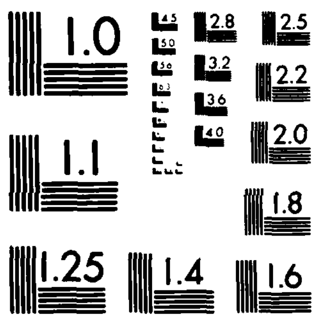
AD-A133 649 A MULTIPLE RANKING PROCEDURE ADAPTED TO DISCRETE-EVENT SIMULATION(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH R T DICKINSON DEC 83 AFIT/CI/NR-83-47D 1/3

UNCLASSIFIED F/G 12/1 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

UNCLASS

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

AD-A133 649

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER AFIT/CI/NR 83-47D	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) A Multiple Ranking Procedure Adapted to Discrete-Event Simulation		5. TYPE OF REPORT & PERIOD COVERED THESIS/DISSERTATION
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Robert Timothy Dickinson		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS AFIT STUDENT AT: The University of Texas at Austin		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS AFIT/NR WPAFB OH 45433		12. REPORT DATE December 1983
		13. NUMBER OF PAGES 181
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASS
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

DTIC
OCT 18 1983
A

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

APPROVED FOR PUBLIC RELEASE: IAW AFR 190-17

27 SEPT 1983

Lynn Wolaver
LYNN E. WOLAVER
Dean for Research and
Professional Development

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

ATTACHED

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASS

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

DTIC FILE COPY

83 10 12 123

83-47

A MULTIPLE RANKING PROCEDURE ADAPTED

TO DISCRETE-EVENT SIMULATION

Publication No. ______

Robert Timothy Dickinson, Ph.D.

The University of Texas at Austin, 1983

Supervising Professor: Dr. James R. Wilson

ABSTRACT

The main objective of this research is to extend the multiple ranking procedure of Dudewicz and Dalal to the case of K normal covariance-stationary processes with unknown and nonidentical covariance structures. To implement this procedure, ~~we~~ author developed a computer program that can be called "on the fly" by an ongoing discrete-event simulation in order to select the best steady-state performance among K alternative policies. For each alternative policy, the proposed support package analyzes the simulation-generated output series to determine if the accumulated sample size is sufficient to meet the predetermined probability requirement (that is, the probability of yielding the correct selection). If the run length is not sufficient, the program determines the number of additional observations to be taken. Upon reaching an acceptable run length, the program reports the estimated mean response of the policy currently under consideration. The run for each alternative policy is executed independently and does not require any information from the runs for other policies.

other policies.

The analysis procedure developed in this research includes subprograms to perform the following operations:

1. The application of a cusum test to detect initialization bias and to truncate the transient portion of the output series.
2. The application of a normality test to ensure that the original data is organized into sufficiently large batches so that the resulting batch means are approximately normal.
3. The application of a spectral method to account for the covariance structure of the batched series by estimating the spectrum at zero frequency.

The ultimate purpose of these subprograms is to yield a simulation-generated time series for which the extended Dudewicz-Dalal procedure is valid.

The final objective of this research is to carry out an extensive experimental validation of the analysis procedure. The systems used for this validation exhibit much diversity in their stochastic behavior and thus provide an indication of the robustness of the multiple ranking procedure.

AFIT/CI/NR 83-47D

AFIT RESEARCH ASSESSMENT

The purpose of this questionnaire is to ascertain the value and/or contribution of research accomplished by students or faculty of the Air Force Institute of Technology (ATC). It would be greatly appreciated if you would complete the following questionnaire and return it to:

AFIT/NR
Wright-Patterson AFB OH 45433

RESEARCH TITLE: A Multiple Ranking Procedure Adapted to Discrete-Event Simulation

AUTHOR: Robert Timothy Dickinson

RESEARCH ASSESSMENT QUESTIONS:

1. Did this research contribute to a current Air Force project?

( ) a. YES ( ) b. NO

2. Do you believe this research topic is significant enough that it would have been researched (or contracted) by your organization or another agency if AFIT had not?

( ) a. YES ( ) b. NO

3. The benefits of AFIT research can often be expressed by the equivalent value that your agency achieved/received by virtue of AFIT performing the research. Can you estimate what this research would have cost if it had been accomplished under contract or if it had been done in-house in terms of manpower and/or dollars?

( ) a. MAN-YEARS ________ ( ) b. $________

4. Often it is not possible to attach equivalent dollar values to research, although the results of the research may, in fact, be important. Whether or not you were able to establish an equivalent value for this research (3. above), what is your estimate of its significance?

( ) a. HIGHLY SIGNIFICANT ( ) b. SIGNIFICANT ( ) c. SLIGHTLY SIGNIFICANT ( ) d. OF NO SIGNIFICANCE

5. AFIT welcomes any further comments you may have on the above questions, or any additional details concerning the current application, future potential, or other value of this research. Please use the bottom part of this questionnaire for your statement(s).

NAME GRADE POSITION

ORGANIZATION LOCATION

STATEMENT(s):

FOLD DOWN ON OUTSIDE - SEAL WITH TAPE

AFIT/NR
WRIGHT-PATTERSON AFB OH 45433

OFFICIAL BUSINESS
PENALTY FOR PRIVATE USE. $300

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 73236 WASHINGTON D.C.

POSTAGE WILL BE PAID BY ADDRESSEE

AFIT/ DAA
Wright-Patterson AFB OH 45433

FOLD IN

A MULTIPLE RANKING PROCEDURE

ADAPTED TO DISCRETE-EVENT

SIMULATION

APPROVED BY SUPERVISORY COMMITTEE:

James R. Wilson

Paul A. Jensen

Robert S. Sullivan

Melba M. Crawford

A MULTIPLE RANKING PROCEDURE
ADAPTED TO DISCRETE-EVENT
SIMULATION

by

Robert Timothy Dickinson, B.S., M.S.

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

December, 1983

In memory of my father, Hal Dickinson.

ACKNOWLEDGMENTS

I would like to express my sincere appreciation to my supervising professor, Dr. James R. Wilson. His guidance, understanding, and encouragement were the solid foundation that supported me during my work. Anv contribution this paper makes to simulation analysis methodology reflects that support.

The lessons Dr. Paul Jensen taught me go beyond the scope of this effort. He is the finest instructor I have ever had. It was a privilege to be in his classes.

I would also like to thank Dr. Melba Crawford and Dr. Robert Sullivan for their interest and help during my months of study. Many times their encouragement made the long hours easier to bear.

Finally to my family, Joyce, Brooke, and Adrienne, without your constant love and help over the years, I know I couldn't have made it. The successful completion of this goal is all the sweeter because I can share it with you.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

83-47

A MULTIPLE RANKING PROCEDURE ADAPTED

TO DISCRETE-EVENT SIMULATION

Publication No. ______

Robert Timothy Dickinson, Ph.D.

The University of Texas at Austin, 1983

Supervising Professor: Dr. James R. Wilson

ABSTRACT

The main objective of this research is to extend the multiple ranking procedure of Dudewicz and Dalal to the case of K normal covariance-stationary processes with unknown and nonidentical covariance structures. To implement this procedure, we develop a computer program that can be called "on the fly" by an ongoing discrete-event simulation in order to select the best steady-state performance among K alternative policies. For each alternative policy, the proposed support package analyzes the simulation-generated output series to determine if the accumulated sample size is sufficient to meet the predetermined probability requirement (that is, the probability of yielding the correct selection). If the run length is not sufficient, the program determines the number of additional observations to be taken. Upon reaching an acceptable run length, the program reports the estimated mean response of the policy currently under consideration. The run for each alternative policy is executed independently and does not require any information from the runs for

other policies.

The analysis procedure developed in this research includes subprograms to perform the following operations:

1. The application of a cusum test to detect initialization bias and to truncate the transient portion of the output series.
2. The application of a normality test to ensure that the original data is organized into sufficiently large batches so that the resulting batch means are approximately normal.
3. The application of a spectral method to account for the covariance structure of the batched series by estimating the spectrum at zero frequency.

The ultimate purpose of these subprograms is to yield a simulation-generated time series for which the extended Dudewicz-Dalal procedure is valid.

The final objective of this research is to carry out an extensive experimental validation of the analysis procedure. The systems used for this validation exhibit much diversity in their stochastic behavior and thus provide an indication of the robustness of the multiple ranking procedure.

identical, he formulated the alternative hypothesis that one of the populations has "slipped" to the right (resulting in a larger mean for that population) relative to the remaining populations. Mosteller's test consists of (a) sorting the overall set of K*n sample observations in ascending order; (b) determining the population with the largest sample value; and (c) counting the number m of observations from that population exceeding all values sampled from the other K-1 populations. If $m \geqslant m_o$, the experimenter rejects the null hypothesis and accepts the hypothesis that the population with the largest sample value has slipped to the right. If $m < m_o$, the null hypothesis is accepted. Mosteller tabulated the size of this test (that is, the probability of Type I error) for different values of K, n, and m_o.

While Mosteller's test has the advantage of being quick and easy to apply, it is not very powerful. Moreover, it requires the same sample size n for every population. The most important aspect of this test, however, is that it is subject to a new type of error. Besides the classical Type I and II errors, there exists the possibility that the null hypothesis is correctly rejected for the wrong reason: the selected population does not have the largest mean. These pitfalls were clearly brought out in Mosteller's original paper. The main contribution of this paper is that it focused attention on a neglected statistical problem of great practical importance. This paper has stimulated extensive research efforts to develop effective testing procedures adapted to a variety of experimental situations.

CHAPTER I

INTRODUCTION

1.1 Genesis of Ranking and Selection Problems

In 1948 Frederick Mosteller published his pioneering work on a statistical question that he called "the problem of the greatest one." Given K populations, Mosteller wanted to select the population with the largest location parameter (for example, the mean or median) by analyzing random samples drawn from each population. The word population was used by Mosteller for a process, $\pi(\theta)$, which generates independent random variables X_1, X_2, ..., X_n, each X_i having the same density function $f(x, \theta)$. A set of X_i's that have been generated by $\pi(\theta)$ is called a random sample from the corresponding population.

The practical importance of such a test is obvious. If several varieties of grain are being tested to determine which variety has the greatest mean yield per acre, the classical approach of the analysis of variance (ANOVA) is not adequate. As a result of applying the F-test of ANOVA, the only possible significant conclusion is that the treatment means are unequal -- and this is frequently known at the outset. The experimenter's real question -- "Which is the best variety of grain?" -- remains unanswered.

Mosteller's approach (1948) was to analyze random samples of fixed size n taken from the K populations using a parameter-free significance test that differs fundamentally from the classical ANOVA approach. Against the null hypothesis that all populations are

1.2 Complications Arising in Simulation

Although Mosteller's procedure was discussed above in the context of an agricultural experiment, it is clear that the procedure can be applied in all branches of experimental science. For many problems which are either too complex to solve analytically or impractical (structurally or economically) to test physically, computer simulation may be the only feasible mode of experimentation. This consideration naturally leads to the use of large-scale system simulations to evaluate and compare alternative policies for system operation. By appropriate analysis of the simulation-generated data sets, the "best" operating policy can be identified. Unfortunately, the following characteristics usually occur in the output series for each alternative:

1. Initialization bias
2. Unknown process variance
3. Unknown autocorrelation structure
4. Marked nonnormality.

These characteristics pose major tactical problems in the execution of a simulation experiment, and they severely complicate any attempt at a follow-up ranking-and-selection analysis.

Initialization bias (Problem 1) occurs when steady-state performance measures are required for a real-world system. Unlike the real-world system, the corresponding simulation model receives intermittent uses over finite periods of simulated time. The

experimenter runs the simulation as needed, records appropriate data, and then shuts the model down. Among the most difficult questions in discrete event simulation are the problems of determining how to start the model and how to obtain measurements that are not biased by the method of starting (Conway, 1963).

Classical methods of statistical analysis are based on independent observations from a single normal population with a known variance. Unfortunately, it has been well documented (Fishman, 1973, 1978; Kleijnen, 1975; Law and Kelton, 1982) that the output responses generated by computer simulations of realistic systems are neither independent (Problem 3) nor normal (Problem 4). Moreover, the variance of the response is usually unknown and unequal across alternative system configurations (Problem 2). While these four problems are not limited to simulated experimentation, they substantially detract from the attractiveness of discrete-event simulation as a method for comparing alternative policies.

Two unique advantages of computer simulation have motivated efforts to overcome these four tactical problems. The first advantage is one that experimentalists always seek to achieve -- perfect homogeneity of the experimental medium. In a simulation experiment, the experimental medium is a sequence of events which describe the activities of the outside world. Since this sequence is a function of the pseudo-random numbers that are sampled to drive the model's exogenous stochastic input processes, the experimenter can reproduce an <u>identical</u> sequence of discrete events whenever such duplication is

desired. The second advantage of simulated experimentation is the ability to perform statistical analysis and control of the simulation as it is being run. After the operation of a discrete-event simulation has been temporarily suspended to perform some type of calculation on the results already obtained, it can be resumed with no loss of information by returning control to the executive time-advance procedure.

1.3 Problem Statement

Given a discrete-event simulation model of a real-world system, the problem is to select the best operating policy from K given alternatives. This requires the development of a multiple ranking procedure that will monitor the relevant output processes generated by each alternative and that will finally yield a correct selection from the set of alternatives with a user-specified level of reliability. The ranking procedure must include effective techniques for handling initialization bias, nonnormality, autocorrelation, and variance estimation in the processes to be analyzed.

1.4 Objectives and Scope of the Research

The main objective of this research is to extend the multiple ranking procedure of Dudewicz and Dalal (1975) to the case of K normal covariance-stationary processes with unknown and nonidentical covariance structures. To implement this procedure, we develop a computer program that can be called "on the fly" by an ongoing

discrete-event simulation in order to select the best steady-state performance among K alternative policies. For each alternative policy, this support package analyzes the simulation-generated output series to determine if the accumulated sample size is sufficient to meet the predetermined probability requirement (that is, the probability of yielding the correct selection). If the run length is not sufficient, the program determines the number of additional observations to be taken. Upon reaching an acceptable run length, the program reports the estimated mean response of the policy currently under consideration. The run for each alternative policy is executed independently and does not require any information from the runs for other policies.

The analysis procedure developed in this research includes subprograms to perform the following operations:

1. The application of the cusum test of Schruben (1982) to detect initialization bias and to truncate the transient portion of the output series
2. The application of the normality test of Shapiro and Wilk (1965) to ensure that the original data is organized into sufficiently large batches so that the resulting batch means are approximately normal
3. The application of the spectral method of Heidelberger and Welch (1981a) to account for the covariance structure of the batched series by estimating the spectrum at zero frequency for that series.

The ultimate purpose of these subprograms is to yield a simulation-generated time series for which the extended Dudewicz-Dalal procedure is valid.

The final objective of this research is to carry out an extensive experimental validation of the analysis procedure. The following systems are used in the validation study:

1. Several sets of autoregressive-moving average processes
2. A set of open, feed-forward queueing networks
3. A set of (s,S) inventory systems.

These experimental vehicles exhibit much diversity in their stochastic behavior and thus provide an indication of the robustness of the multiple ranking procedure.

1.5 Organization of the Dissertation

Chapter II provides a survey of the literature on the following topics: (a) ranking-and-selection procedures, (b) tests for initialization bias, (c) tests for normality, and (d) spectral analysis of simulation output. Chapter III presents the development of the extended Dudewicz-Dalal procedure together with all of its required support routines. Chapter IV contains a tabulation and analysis of the results of the experimental validation study. A précis of the main findings of this research is given in Chapter V along with recommendations for future work.

CHAPTER II

LITERATURE REVIEW

2.1 Multiple Ranking Procedures

After Mosteller's work, the next significant step in attempting to solve "the problem of the greatest one" was completed by Bahadur (1950). In his work Bahadur reiterated the fact that the usual statistical theory for testing hypotheses of the form

$$H_o: \mu_1 = \mu_2 = \cdots = \mu_K$$

is inadequate where a definite course of action is required to select the best population. He proceeded to consider explicitly the purpose of selection and the loss involved in making any particular erroneous selection. Using the same definition of population and the same goal as Mosteller, Bahadur proved that selecting the population with the largest sample mean $\bar{X}_i$ is the optimal procedure under the following conditions:

1. Impartial selection is required.
2. The experimenter seeks to maximize the probability of correctly selecting the population with the optimal expected value.
3. The basic observations are independent normal random variables.
4. The sample sizes are equal.

For other unknown parameters which are to be compared, Bahadur cites necessary and sufficient conditions that must be met to ensure the

following property: If X_1 and X_2 are independent estimates of the corresponding unknown parameters C_1 and C_2, and in a given instance $X_1 > X_2$, then it is more reasonable to suppose that $C_1 > C_2$ than to suppose that $C_1 < C_2$. It was this basic work in the ranking of unknown parameters based on random samples that lead J.P.C. Kleijnen (1975) to state, "Historically, multiple ranking procedures can be traced back to the work of Bahadur in 1950."

After the pioneering work of Mosteller and Bahadur, procedures to solve "the problem of the greatest one" branched into three separate approaches. These three methods of ranking alternatives are classified as follows:

1. Indifference Zone Approach
2. Subset-selection Approach
3. Sequential techniques.

All three methods come under the general heading of ranking - and - selection procedures or multiple ranking procedures (MRP). It is this latter nomenclature that will be used throughout this dissertation.

2.1.1 Indifference Zone Approach

Although the groundwork was laid by Mosteller and Bahadur, it was R. E. Bechhofer's effort in the MRP area which represented a major event in statistical thought (Dudewicz, 1976). Bechhofer (1954) developed a procedure for determining the number of observations required for selecting the single "best" population (that is, the population with the largest mean value) from K competing populations

based on a predetermined probability of correct selection. While Bahadur justified using the sample means to compare the true means of normal populations, Bechhofer specified how large a sample size would be required to ensure a desired probability, P^*, that the correct population is chosen by the test procedure. Bechhofer's procedure also allowed the experimenter to select an "indifference zone" which effectively prevented the experimenter from taking large samples to detect only small differences in sample means. Bechhofer's problem statement can be summarized as follows:

There exist K populations $\pi_i (i=1, 2, \ldots, K)$, where

$$\mu_{[1]} < \mu_{[2]} < \cdots < \mu_{[K-1]} < \mu_{[K]}$$

represent the ranked means. The objective is to select the population with the largest mean $\mu_{[K]}$.

Bechhofer's procedure specifies the size n of the random sample to be taken from each population so that by choosing the population with the largest sample mean, the probability of a correct selection (CS) is greater than or equal to a predetermined constant P^*. Since large samples are required to select the best population if the population means differ only slightly (and the loss involved in a wrong selection is then small), the requirement for a floor on Pr(CS) is necessary only if the best population mean is at least a specified number of units, Δ^* ($\Delta^* > 0$), better than the next best mean. Symbolically, this requirement has the form

$$\Pr(CS) > P^* \text{ if } \mu_{[K]} - \mu_{[K-1]} \geqslant \Delta^* \qquad (2.1.1)$$

where Δ^* indicates the size of the "indifference zone". In his

original work, Bechhofer concentrated on a single-sample method for selecting the best population; and he assumed that all observations are independent and are taken from normally distributed populations with a known common variance. Bechhofer also extended his procedure to the selection of the population with the smallest mean and to the selection of the populations with the m "best" (largest or smallest) means ($m < K$).

In determining the proper sample size, Bechhofer introduced the concept of the "least favorable configuration" (LFC). Bechhofer stated that the probability requirement (2.1.1) must be valid when the K unknown population means are arranged in a configuration that makes it the most difficult to distinguish the best of the population means. He showed that this LFC occurs when:

$$\mu_1 = \cdots = \mu_{[K-1]} = \mu_{[K]} - \Delta^*$$

If the means do not fall in this configuration, then the actual probability of correct selection exceeds the floor P^*.

In a simplified derivation of the formula for the common sample size n given P^* and Δ^*, Barr and Rizvi (1966) show that solving equation (2.1.2) for h and then substituting the result into equation (2.1.3) determines the required sample size:

$$P^* = \int_{-\infty}^{\infty} \Phi^{K-1}(x + h)\, d\,\phi(x) \qquad (2.1.2)$$

where

$1/K < P^* < 1$, Φ is the standard normal distribution function, and

$$n = (\sigma h/\Delta^*)^2 \qquad (2.1.3)$$

and where σ = known common standard deviation of all K populations. Bechhofer (1954) tabulated values of h in his original paper. Obviously h and n increase as K and P^* increase. In a paper published 15 years later, Dudewicz (1969) relieved the experimenter of having to solve (2.1.2) or having to use tables to find the appropriate h value. An accurate computing formula for h was shown to be

$$h(P^*,K) = 2\ [-\ n(1-P^*)]^{1/2} \qquad (2.1.4)$$

While this breakthrough by Bechhofer allowed the experimenter to predetermine P^* and Δ^*, it had a major drawback in requiring a known common variance for all populations. Bechhofer, Dunnett and Sobel (1954) subsequently developed a two-sample procedure for ranking K normal populations with unknown variances

$$\sigma_i^2 = a_i\,\sigma^2 \qquad \sigma_i = 1, 2, \ldots, K,$$

where σ^2 is unknown but the a_i's are known. This procedure also requires calculation of a critical value h in a manner similar to Bechhofer's first paper. The two-sample procedure is:

1. Take an initial sample of $a_i N_o$ observations from the i^{th} population π_i ($i = 1, 2, \ldots, K$).
2. Calculate the mean square error,

$$S_o = V^{-1} \sum_{i=1}^{k} a_i^{-1} \sum_{j=1}^{N_o} (X_{ij} - X)^2$$

This is an unbiased estimate of σ^2 with

$$V = N_o \sum_{i=1}^{k} a_i - K$$

degrees of freedom.

3. Take a second sample of $(N - N_o)a_i$ observations from each of the π_i $(i = 1, 2, \ldots, K)$ populations, where

$$N = \max \{N_o, [2 S_o^2 (h_n / \Delta^*)^2]\} \qquad (2.1.5)$$

In equation (2.1.5), $[\cdot]$ denotes the greatest integer function, and $h = h(N_o, K, P^*)$ is obtained from Table 3 of Dunnett and Sobel (1954). If N equals N_o, a second sample is unnecessary.

4. Calculate for each π_i the overall sample mean

$$\bar{X}_i = (a_i N)^{-1} \sum_{j=1}^{a_i N} X_{ij} \qquad (i = 1, 2, \ldots, K)$$

Denote the ranked values of X_i by

$$\bar{X}_{[1]} < \bar{X}_{[2]} < \ldots < \bar{X}_{[K]}.$$

5. Rank the μ_i according to the ranking of the observed X_i -- that is, select the population which gave rise to $\bar{X}_{[K]}$ as the "best" population.

C. W. Dunnett (1960) considered the situation of ranking K normal populations with unknown means, equal variances and covariances, and some a priori information about the distribution of the unknown population means. Specifically, the unknown population means are themselves normally distributed. A typical problem to which this model could apply is the selection of the best of K varieties of grain which have been chosen at random from the same parent population of grain varieties. Another extension developed by Dunnett applies to the case where there is prior knowledge about the values of the population means. This a priori information is used to justify a

smaller sample size than is dictated by Bechhofer's procedure because it is known that the LFC does not occur. Here again Dunnett recognized the possibility of various definitions of "best," depending upon the requirements of the particular application. He assumed, as did his predecessors, that there is a single characteristic by which the various experimental populations are to be judged, and that the best population is the one which possesses on the average the highest value of this characteristic. Dunnett's procedure has not been widely used because certain multivariate normal integrals required by the procedure have not been tabulated. This procedure is also encumbered by the questionable accuracy of the a priori information on which it is based.

Somerville (1970) also approached the idea of ranking alternatives through the use of a probability requirement and an indifference zone, but he tied both P^* and Δ^* to economic costs. His procedure determined the "optimum" sample size for choosing the population having the best (largest or smallest) mean when a specified economic loss is suffered if an incorrect decision is made. In this situation, the expected loss due to an incorrect decision must be balanced against the expected cost of experimentation. Somerville arrived at the same LFC as Bechhofer; but he used the minimax principle, with Δ^* and P^* being determined by economic costs instead of being specified by the experimenter. His basic assumptions (normality, known variances, independent observations) coincided with those of Bechhofer.

Chambers and Jarrett (1964) also followed the path of Bechhofer in deciding how large a sample should be taken from each of K populations in order to give at least a specified probability of selecting the best population when the indifference zone has width Δ^*. However, their double sampling procedure is designed to select the best of K nonnormal populations when the population variances $\{\sigma_i^2\}$ depend on the corresponding unknown means $\{\mu_i\}$; and the form of this relationship

$$\sigma_i = \sigma(\mu_i)$$

is known and common to each population. Binomial and Poisson populations provide examples of this situation. The Chambers and Jarrett procedure is shown to be valid only for large samples. For example, they used an initial sample size of 750 for selecting the best of 3 binomial populations with $P^* = 0.95$ and $\Delta^* = 0.02$.

Chambers and Jarrett's derivation of the proper sample size closely follows the analvsis given by Barr and Rizvi (1966) for the Bechhofer procedure. (See equation (2.1.2).) The recommended procedure is: (assuming "best" as smallest mean)

1. An initial sample of size N_o is taken from each population, π_i ($i = 1, 2, \ldots, K$).
2. The smallest sample mean is used to estimate the smallest population mean $\mu_{[1]}$.
3. The total sample size N needed from each population is estimated by inserting the estimated $\mu_{[1]}$ into equation (2.1.6):

$$P^* = \int_{-\infty}^{\infty} \phi(u) \, [1 - \Phi(\frac{u - \delta^*}{\gamma^*})]^{K-1} \, du, \qquad (2.1.6)$$

where ϕ and Φ respectively denote the standard normal density and distribution functions, and where

$$\gamma^* = \frac{\sigma(\mu_{[1]} + \Delta^*)}{\sigma(\mu_{[1]})}, \qquad \delta^* = \frac{\sqrt{n}\,\Delta^*}{\sigma(\mu_{[1]})}$$

4. An extra sample of size $N - N_0$ should be taken from each population (none if $N < N_0$).
5. The population yielding the smallest final sample mean is then selected as the "best" population.

One advantage of this two-stage procedure is that it allows Δ^* to be specified as a *percentage of the smallest* population mean rather than as an absolute quantity. Again following the form of Bechhofer's original work, Chambers and Jarrett discussed the idea of a LFC and they produced tables that specify values of δ^*.

Up to this point, all MRPs required some prior knowledge of the value of the population variance. Either the actual value of the variance, known variance ratios, or a known functional relationship with the unknown mean had to be available to the experimenter before any decisions could be made. The next logical step in the evolution of MRPs was taken by Dudewicz and Dalal (1975) — namely, the development of a procedure which solves the general ranking problem with $\sigma_i^2 (i = 1, 2, \ldots, K)$ unknown and unequal. They considered the same problem addressed by Bechhofer -- including the concepts of a

probability requirement P^*, an indifference zone Δ^*, and a *LFC*, but without the assumption of known and equal variances. For this situation, Dudewicz and Dalal developed the following two-stage procedure, P_E, to determine the required sample size n_i for the i^{th} population:

1. Take an initial sample $\{X_{i,1}, \ldots, X_{i,n_o}\}$ of size $n_o(>2)$ from $\pi_i(i = 1, 2, \ldots, K)$ and define

$$\bar{X}_i(n_o) = \sum_{j=1}^{n_o} X_{i,j}/n_o \qquad (2.1.7)$$

$$S_i^2 = \sum_{j=1}^{n_o} (X_{i,j} - \bar{X}_i(n_o))^2/(n_o - 1) \qquad (2.1.8)$$

$$n_i = \max\{ n_o + 1, [(S_i h/\Delta^*)^2] \} \qquad (2.1.9)$$

where $[z]$ denotes the smallest integer $\geqslant z$ and $h = h_{n_o}(K, P^*)$ is the unique solution of

$$\int_{-\infty}^{\infty} \{[F_{n_o}(z+h)]^{K-1}\} f_{n_o}(z)dz = P^* \qquad (2.1.10)$$

where $F_{n_o}(\cdot)$ and $f_{n_o}(\cdot)$ are respectively the distribution and density function of a Student's -t random variable with $n_o - 1$ degrees of freedom.

2. Take $n_i - n_o$ additional observations

$X_{i,n_o+1}, \ldots, X_{i,n_i}$ from $\pi_i(i = 1, 2, \ldots, K)$

and define

$$\tilde{X}_i = \sum_{j=1}^{n_i} a_{ij} X_{ij} \qquad (2.1.11)$$

where the a_{ij}'s $(i = 1, 2, \ldots, K;\ j = 1, \ldots, n_i)$ are chosen

so that

$$\sum_{j=1}^{n_i} a_{ij} = 1 \qquad (2.1.12)$$

$$a_{i1} = \cdots = a_{in_o} \qquad (2.1.13)$$

$$S_i^2 \cdot \sum_{j=1}^{n_i} a_{ij}{}^2 = (\Delta^*/h)^2 \qquad (2.1.14)$$

(Note: For K < 3 this weighting scheme is not required. The mean of n_i observations is used for comparison.)

3. Rank the populations based on the $\tilde{\tilde{X}}_i$ values where $\tilde{\tilde{X}}_{[1]} < \tilde{\tilde{X}}_{[2]} < \cdots < \tilde{\tilde{X}}_{[K]}$ and select the population which yields $\tilde{\tilde{X}}_{[K]}$ (largest sample mean).

The justification for this procedure, as presented in the paper, shows that the method is independent of the σ_i^2 ($i = 1, 2, \ldots, K$) values and that $\Pr(CS \mid P_E, LFC) = P^*$. Extensive P^* tables are provided in the paper for varying values of K, n_o, and h.

In a previous paper, Dudewicz (1971) showed that if the σ_i^2 are not known, no single-stage sampling procedure can satisfy the probability requirement (2.1.1). While Dudewicz admitted that there might be some resistance to a multi-stage procedure, he emphasized two important points: (a) double sampling plans achieve almost the same efficiency increase over a single sampling as that achieved by a fully sequential plan; and (b) in most cases, the first stage of a double sampling plan is actually equivalent to running a pilot study for preliminary variance estimation prior to carrying out a fully sequential procedure (see also Dudewicz, Ramberg, and Chen, 1975).

Dudewicz, Ramberg, and Chen (1975) also presented a Procedure $\mathcal{P}$ that is equivalent to the previously shown P_E but is more amenable to machine computation. Procedure $\mathcal{P}$ is:

1. Complete step 1 of P_E

2. Take $n_i - n_o$ additional observations $X_{i,n_o+1}, \ldots, X_{i,n_i}$ from π_i and calculate the new sample mean

$$\bar{X}_i'(n_i - n_o) = (n_i - n_o)^{-1} \sum_{j=n_o+1}^{n_i} X_{ij}, \qquad (2.1.15)$$

the weights

$$W_i = (n_o/n_i)[1+([n_i(hS_i/\Delta^*)^{-2} - 1](n_i - n_o)/n_o)^{1/2}] \qquad (2.1.16)$$

$$W_i' = 1 - W_i\ , \qquad (2.1.17)$$

and the final weighted mean

$$\tilde{X}_i = W_i\bar{X}_i(n_o) + W_i'\bar{X}_i'(n_i-n_o) \quad \textit{for } i=1, \ldots, K. \qquad (2.1.18)$$

3. Complete step 3 of P_E.

Dudewicz, Ramberg, and Chen (1975) presented the numerical analysis that they used to compute extensive tables of the critical value h as a function of n_o, K, and P^*. The solution procedure for equation (2.1.10) involves a nonsequential search over a grid of h values. For each value of h, a 128-point Gauss-Legendre quadature formula is used to approximate the required integral over 9 laboriously determined subintervals. In Chapter III, we develop a solution procedure which is completely automated and is therefore suitable for use in discrete-event simulation.

As was the case for the original Bechhofer paper in 1954, both procedures P_E and $\mathcal{P}$ can be generalized to select the population with

the m best means ($1 \leqslant m < K$). It is important to note that the extended selection procedure will not indicate that the m selected populations are ranked or ordered in any way among themselves; the probability requirement only refers to the event in which the unordered set of m selected populations is the same as the unordered set of the m best populations. This particular selection goal might be useful if it is decided to identify several good options, since the best population might prove unacceptable for other reasons. The procedure given by Bechhofer (1954) could be used to select the m best populations and to specify the proper ordering of all m populations.

For the simulator, both these procedures, P_E and $\mathcal{P}$, were most welcome. To be able to finally shake the bounds imposed by having to assume some known property of the population variance greatly increased the practical value of MRPs. This is borne out by the following statement in Law and Kelton (1982): "Assuming known or equal variances is very unrealistic in simulation."

The last major hurdle in using MRP to analyze discrete-event simulation is the presence of serial correlation in the output series. Dudewicz and Zaino (1977) chose to model the observations from such a process π_i ($i = 1, 2, \ldots, K$) bv an autoregressive scheme of order 1:

$$X_{in} = \rho X_{i,n-1} + Z_{in} , \qquad (2.1.19)$$

where $|\rho| < 1$, and $\{Z_{in}: n \geqslant 1\}$ is a sequence of uncorrelated random variables with mean $(1 - \rho)\mu_i$ and variance σ^2. For equation (2.1.19), we have:

$$E(X_{in}) = \mu_i \qquad (2.1.20)$$

$$\mathrm{Var}(X_{in}) = \sigma_x^2 = \sigma^2/(1-\rho^2) \tag{2.1.21}$$

$$R_s = \mathrm{Cov}(X_{in}, X_{i,n+s}) = [\sigma^2/(1-\rho^2)] \cdot \rho^{|s|} = \sigma_x^2 \rho^{|s|} \tag{2.1.22}$$

Using this AR(1) model, Dudewicz and Zaino showed that the proper sample size, N_3, to use to compensate for the known covariance structure (2.1.22) is found by taking N_3 to be the smallest integer satisfying

$$\frac{1}{N_3}\left\{\frac{(1+\rho)}{(1-\rho)} - \frac{2\rho(1-\rho^{N_3})}{N_3(1-\rho)^2}\right\} < \frac{1}{N}, \tag{2.1.23}$$

where N is the sample size required by Bechhofer's (1954) procedure in the case that $\rho = 0$. The onlv deviation from Bechhofer's basic assumptions are:

1. Δ^* is specified as a percentage of σ_x, equation (2.1.20);
2. The population sample observations, $X_{i,j}(j = 1, 2, \ldots, n)$, are not independent; and
3. All ρ_i $(i = 1, \ldots, K)$ are equal.

A good approximation to N_3 is given by

$$N_2 = N\,\frac{1+\rho}{1-\rho} \tag{2.1.24}$$

for values of $\rho > -0.5$. The authors also present several graphs that show how the required sample size grows as the value of $|\rho|$ increases. This fact is intuitively obvious since as $|\rho|$ increases, a fixed number of samples will yield "less" information about the population mean.

Dudewicz and Zaino extended their procedure to the ..tuation

in which the ρ_i values are unknown and unequal. Finally they considered the situation in which both σ_i^2 and ρ_i are unknown and unequal. For this latter condition, the Dudewicz and Zaino heuristic procedure is:

1. Take an initial sample size of $N_o = 30$ from each simulation model (population). Calculate the number of observations which would be needed if $\rho_i = 0$:

   $$M_i = \max(N_o, [(S_i h/\Delta^*)^2]) , \qquad (2.1.25)$$

   where $(S_i h/\Delta^*)$ is calculated from step 1 of the Dudewicz and Dalal (1975) procedure. If $M_i > N_o$ take $M_i - N_o$ additional observations.

2. Using all M_i observations, calculate the sample mean $\bar{X}_i$ and the sample lag-one correlation coefficient

   $$\hat{\rho}_i = \frac{\sum_{n=2}^{M_i} (X_{in} - \bar{X}_i)(X_{i,n-1} - \bar{X}_i)}{\sum_{n=1}^{M_i} (X_{in} - \bar{X}_i)^2} . \qquad (2.1.26)$$

   Form the $100(1 - \alpha)$% confidence interval for ρ_i from

   $$(\rho_i - \hat{\rho}_i)^2 \leqslant \frac{M_i - 1}{M_i(M_i - 3)} (1 - \hat{\rho}_i^2)\, t_{M_i - 3}(1 - \alpha/2) , \qquad (2.1.27)$$

   where $\alpha = .05$ and $t_r(q)$ is the 100q percent point of Student's - t distribution with r degrees of freedom. If the interval (2.1.27) contains $\rho_i = 0$, judge M_i to be an adequate sample size for population i and go to step 4.

3. If the interval from step 2 does not contain $\rho_i = 0$,

calculate

$$N_{2i} = M_i \frac{1 + \rho_i}{1 - \rho_i} \qquad (2.1.28)$$

and take $N_{2i} - M_i$ additional observations from population i.

4. Using all the observations taken from population i, compute the overall sample mean $\bar{\bar{X}}_i$ and select that population which produces the largest $\bar{\bar{X}}_i (i = 1, 2, ..., K)$.

Two points should be emphasized about this scheme. The first is that all the Dudewicz and Zaino procedures are valid only if an AR(1) process accurately models the output data. Secondly, only lag-one correlation is taken into account. It is still assumed that the observations across alternatives are independent, i.e. $X_{i,j}$ is independent of $X_{i+\ell,j}$ ($\ell = 1, 2, ..., K-1$).

2.1.2 Subset Selection Approach

Another method of comparing alternatives was pioneered by Gupta (1956). In the subset selection approach, the goal is to select a *non-empty subset* of the populations so as to include the best population. In this approach, the size of the selected subset is not fixed in advance, but is determined by the observations themselves. For the problem of K normal populations with unknown means and a common known variance, the procedure developed by Gupta (1956) selects the population that yields $\bar{X}_i$ if and only if

$$\bar{X}_i > (\max_{1 \leq j \leq K} \bar{X}_j) - d\, \sigma/n^{1/2} \qquad (2.1.29)$$

where σ = common known population variance
n = common sample size
$d = d(K, P^*) > 0$ is the solution to

$$P^* = \int_{-\infty}^{\infty} \Phi^{K-1}(t+d)d \cdot \phi(t)\, dt, \qquad (2.1.30)$$

where Φ is the cumulative distribution function of a standard normal variable. Work has also been done by Gupta to handle common unknown variances and unequal variances. The subset selection approach differs from the indifference zone technique in that the latter requires specification of two constants P^* and Δ^* to select a fixed number, m, of "best" populations, while the former only requires P^* to pick a random-sized set containing the best population.

Kleijnen, Naylor and Seeks (1972) have suggested that the subset selection procedure can be used to reduce the number of possible alternatives when K is large; then the indifference-zone technique can be applied to the remaining alternatives to find the "best" population. This approach is onlv appropriate if the experimenter is seeking a single "best" population (m=1). This approach will not work for $m \geqslant 2$ because there is no way to ensure that every population whose mean falls within Δ^* units of the "best" population mean will be included in the subset selected by Gupta's procedure. Missing some nearly optimal populations can be quite damaging if those populations possess a secondary attribute which makes them more desirable than the "best" population.

2.1.3 Sequential Approach

Both of the previously discussed MRP approaches — subset

selection and indifference zone — are either single-stage or two-stage procedures. Starr (1966) showed that it is more efficient for the experimenter to take observations one at a time and terminate the experiment based on a known stopping rule as soon as the desired goal is reached. Sequential MRPs have this trait.

Bechhofer (1958) developed the following sequential procedure for finding the "best one of several normal populations with a common known variance":

1. At the m-th stage of experimentation ($m = 1, 2, \ldots$), take an observation from each of the K populations. Starting with $m = 2$, compute the stopping statistic $Z_m(d_m)$. For complete details on the form of the stopping statistic, see Bechhofer (1958).
2. If $Z_m(d_m) \leqslant (1 - P^*)/P^*$, stop experimentation and select the population with the largest sample mean.
3. If $Z_m(d_m) > (1 - P^*)/P^*$, take another observation from each of the populations, replace m by m+1, and go to step 2.

Due to the tedious computations required at each stage to check the stopping rule, Bechhofer and Blumenthal (1962) devised a new computing formula for $Z_m(d_m)$.

Paulson (1964) developed an alternative to Bechhofer's sequential procedure which is substantially easier to use. Paulson's procedure includes the ability to eliminate certain populations from further sampling once they were identified as "inferior". This elimination is referred to as "taking advantage of a more favorable

configuration (MFC)." As previously mentioned, the probability requirement is satisfied even if the population means are in an LFC. It is clear that as the means depart from the LFC to yield a MFC, a smaller sample size will suffice.

The Paulson procedure assuming a common known variance is:

1. Select $\lambda = \Delta^*/4$ and calculate

$$a_\lambda = [\sigma^2/(\Delta^* - \lambda)] * \ln((K-1)/(1-P^*)) \qquad (2.1.31)$$

2. Take one observation from each population $(X_{11}, X_{21}, \ldots, X_{K1})$. Eliminate any population π_i for which $X_{i1} < \max\{X_{11}, X_{21}, \ldots, X_{K1}\} - a_\lambda + \lambda$. If all but one population is eliminated, select it as the best and stop. Otherwise go to step 3.

3. Take another observation from each population not eliminated in step 2. Proceeding by induction, at the r^{th} stage we eliminate any population π_i for which

$$\sum_{s=1}^{r} X_{is} < \max\{\sum_{s=1}^{r} X_{vs}: \text{ all remaining } v\} - a_\lambda + r\lambda . \qquad (2.1.32)$$

The limit on r equals W_λ, where W_λ is the greatest integer in a_λ / λ. As soon as K - 1 populations are eliminated, the remaining population is selected as best. If after W_λ stages there is more than one population remaining, go to step 4.

4. The experiment is terminated at stage $(W + 1)$ by selecting population i for which

$$\sum_{s=1}^{W_\lambda +1} X_{is}$$

is the greatest. Paulson showed that the probability requirement (2.1.1) is satisfied by this procedure and bv a similar method for the case of common unknown σ^2. Kleijnen (1974) reported on Monte Carlo experiments showing that Paulson's procedure performs better than any other comparable single stage multi-stage, or sequential procedure.

Although the sequential procedures are more efficient for a given P^* and Δ^*, their application by simulators has been limited. This is partially due to the fact that no sequential procedure addresses the problem of unknown variances or correlated observations; however, the main problem is that it is very cumbersome to simulate several systems in parallel.

2.2 Initialization Bias

As previously mentioned the objective of many computer simulation experiments is to select the best alternative operating policy for a real-world system based on simulation-generated estimators of steady-state performance under each policy. For example, a proposed set of inventory reorder points and order quantities may be used as decision variables for a model of an inventory system with a particular demand distribution. The relative effectiveness of these proposed policies can then be measured by comparing the simulated average monthly costs. In such a case, the initial (starting) conditions (amount on hand, amount on order) can

seriously affect both the bias and the variance of the simulation-generated cost estimators.

Discrete-event simulation of such a stochastic system requires that starting conditions for each run be completely specified. Ideally, these initial conditions should be randomly selected from the equilibrium state probability distribution for the system. However, an experimenter who has enough information to do this has no need to execute the simulation. The more common situation is that the experimenter has some basic understanding of the system that he has garnered from past testing or analysis. Therefore, he must pick the most realistic initial conditions possible based on his a priori knowledge of steady-state system operating characteristics. If no reliable information is available, "empty and idle" is always a possible starting state. In the inventory example, this corresponds to an empty warehouse with no stock on order. Any computer simulation that begins with such a sequence of unusual events -- that is, events having a low probability of occurring under normal operating conditions -- will generate output that is contaminated by initialization bias (Schruben 1982). This initialization bias can be a major source of error in estimating a steady-state system performance measure.

Since the specified purpose of our simulation experiments is to obtain the correct sample sizes so that the K alternatives can be ranked based on the sample means, a method is needed to eliminate any bias in these estimators caused by improper starting conditions. One

method to overcome this initialization bias problem is to allow a sufficiently long computer run so that the initial condition effects are negligible. Even though these effects typicallv decay geometrically, convergence to steady-state conditions can still be quite slow (Conway, 1963). Therefore, such a method can be costly in that the required sample sizes can be prohibitively large.

The usual method of controlling simulation initialization bias is to allow the model to run for a "warm-up" period before output data are collected. This allows those observations which are the most "contaminated" bv the choice of starting conditions to be discarded. As a consequence, the bias of the estimated steady-state mean response is reduced. This procedure is referred to as output truncation, and the time index of the last observation to be discarded is called the truncation point.

There are, however, several difficulties with truncating data from the beginning of each run. If too few observations are truncated, the remaining bias adversely affects the results. Discarding an excessive amount of data is not onlv asteful but also increases the variance of cumulative statistics like the sample mean.

A comprehensive review of previously proposed "truncation rules" is presented by Wilson and Pritsker (1978a). Many of these procedures are heuristic rules of thumb which specify the truncation point beyond which data are not significantly distorted by the initial conditions. In a follow-up paper, Wilson and Pritsker (1978b) developed a generalized procedure for evaluating startup policies with

associated truncation rules, and they used this procedure to test many of the methods reported in their first paper. They specifically pointed out that "The truncation rules of thumb examined in this research are very sensitive to parameter misspecification, and their use can result in excessive truncation".

Before considering the proper way to select a truncation point, we must first determine if initialization bias is present in the output series generated by a simulation. Schruben (1979) developed a two-sided statistical test for the presence of initialization bias based on cumulative sum (cusum) statistics. He chose the cusum statistic because of its demonstrated sensitivity in industrial quality control applications. To illustrate cusum techniques, consider the output series $X_1, \ldots, X_n$. The jth cusum (S_j) is:

$$S_j = \sum_{i=1}^{j} (X_i - \mu_o) \qquad j = 1, 2, \ldots, n \qquad (2.2.1)$$

where μ_o is the process mean, and the X_i are independent normal variates with $E(X_i) = \mu_o$, Var $(X_i) = \sigma_o^2$ so that

$$E(S_j) = 0$$

$$Var(S_j) = j\,\sigma_o^2$$

There are three problems that arise in the application of cusum tests to determine initialization bias: (1) the output series is usually correlated; (2) the process mean, μ_o, is not known; and (3) the observations, X_i, are not normally distributed. The question of how to address correlated data is addressed first.

If the output data series is correlated but μ_o is known, it has been shown that the normalized cusum

$$S_j^o = S_j/(\sigma j^{1/2}), \qquad j = 1, 2, \ldots, n \tag{2.2.2}$$

converges in distribution to standard Brownian motion as $n \longrightarrow \infty$ (Schruben 1979). In this case, the correlation is accounted for by using as the measure of the variance, σ^2, the value

$$\sigma^2 = \sum_{\ell=-\infty}^{\infty} \gamma_\ell \tag{2.2.3}$$

where γ_ℓ denotes the autocovariance at lag ℓ:

$$\gamma_\ell = E\,[(X_i - \mu_o)(X_{i+\ell} - \mu_o)] \tag{2.2.4}$$

To adapt this result to a discrete-event simulation in which μ_o and σ^2 are unknown, Schruben suggested estimating σ^2 from the portion of the data that is "safe" from the effects of any initial bias. He therefore recommended that only the last half of the data be used to estimate σ^2. The entire output series is grouped into b equal-size adjacent and nonoverlapping batches of observations, and the series of batch means $(\bar{X}_k : k = 1, \ldots, b)$ is indexed in reverse order so that $\bar{X}_1$ and $\bar{X}_b$ respectively represent the last and first such observations generated by the simulation. There are $s = b/2$ "safe" observations $(\bar{X}_k : k = 1, \ldots, s)$ that can be treated as IID normal variates. For the <u>batch mean process</u>, the parameter σ^2 is estimated by

$$V_s^2 = (s-1)^{-1} \sum_{k=1}^{s} (\bar{X}_k - \bar{\bar{X}}_s)^2 \tag{2.2.5}$$

where

$$\bar{\bar{X}}_s = s^{-1} \sum_{k=1}^{s} \bar{X}_k \qquad (2.2.6)$$

is the grand mean of the "safe" data.

Since μ_o is unknown, Schruben developed an alternative to the test statistic (2.2.2).

Let

$$\bar{\bar{X}}_b = b^{-1} \sum_{k=1}^{b} \bar{X}_k \qquad (2.2.7)$$

$$\hat{S}_\ell = \sum_{k=1}^{\ell} (\bar{X}_k - \bar{\bar{X}}_b), \quad \ell = 1, \ldots, b, \qquad (2.2.8)$$

$$S^* = \max \{ |\hat{S}_\ell / (b^{1/2} \hat{\sigma})| : \ell = 1, \ldots, b \}. \qquad (2.2.9)$$

Using the values computed from equations (2.2.5) through (2.2.9), Schruben proved that in the absence of initialization bias, the event

$$S^* \leqslant g(\alpha) = [-0.5 * \ln(\alpha/2)]^{1/2} \qquad (2.2.10)$$

has asymptotic probability $1 - \alpha$ as $b \longrightarrow \infty$. Thus for a prespecified level, α . of Type I error, equation (2.2.10) can be used to construct a two-sided statistical test for initialization bias with rejection of the hypothesis of insignificant initialization bias when:

$$S^* > g(\alpha).$$

The most recent work in the area of detecting initialization bias (Schruben 1982) describes a one-sided test related to (2.2.9). It should be noted that in the absence of initialization bias, the process $\{\hat{S}_\ell / (b^{1/2} \hat{\sigma}) : \ell = 1, \ldots, b\}$ behaves asymptoticallv as a

standard Brownian bridge -- i.e., Brownian motion on the unit interval conditioned to start at and return to zero Let m denote the value of the index ℓ at which $\hat{S}_\ell$ attains its minimum, and let $\hat{t} = m/b$. Schruben showed that if no initial bias is present, then the statistic

$$\hat{g} = (\hat{S}_m^2/b)/[3\,\hat{\sigma}^2\,\hat{t}(1-\hat{t})] \qquad (2.2.11)$$

will have approximatelv an F distribution with 3 and b/2 degrees of freedom. The hypothesis of no initialization bias is rejected if the computed significance probability for the F-ratio (2.2.11) is less than a prespecified level of Type I error.

Schruben states that this new test is more powerful than his previous procedure since it is a one-sided test. He assumes that the user is looking for negative initial bias, the typical case for simulations started "empty and idle". If positive bias is suspected, the required adjustment is to multiply the data by -1 and proceed as before.

In this same article, Schruben presented a modification to the procedure just described which does not require estimation of the scale parameter σ^2. The complete procedure is:

1. Compute (2.2.11) using the first half of the simulation output data and set $\hat{\sigma}^2 = 1$; call the result $\hat{g}_f$.
2. Compute (2.2.11) using the last half of the simulation output data and set $\hat{\sigma}^2 = 1$; call the result $\hat{g}_\ell$.
3. Let $F_{3,3}(1-\alpha)$ denote the $1-\alpha$ quantile of the F-distribution with 3 and 3 degrees of freedom. Reject the hypothesis of no negative initialization bias if

$$\hat{g}_f / g_\ell > F_{3,3}(1 - \alpha). \tag{2.2.12}$$

Once we have found an adequate test to determine if a sequence of simulation-generated data has initial bias, there is still the problem of how much data to delete if bias is found. Such a situation is addressed by Heidelberger and Welch (1982). Using Schruben's test (2.2.10) for initialization bias, they proposed a sequential truncation procedure. If the hypothesis of no initial bias is rejected, the first 10% of the data is deleted and the test is rerun. They found that no severe penalties were incurred when there was no initial transient (bias); and in almost all cases where there was bias, the truncated sample mean provided a better estimator of the steady-state mean than the untruncated sample mean.

2.3 Normality

As previously referenced, Dudewicz and Zaino have developed the onlv multiple ranking procedure that handles correlated data; however, this procedure still requires that the data are normally distributed. Dudewicz and Zaino lightly pass over this requirement as if it were one that occurs routinelv in discrete-event simulations. In actuality, the assumption of normality is frequently violated.

The first step that must be addressed when trying to induce normality is an adequate statistical test for normality. Many such tests exist. Fishman (1978) cites numerous studies showing the Shapiro-Wilk (SW) test to be the most powerful test for normality currently available. In particular, Shapiro, Wilk and Chen (1968)

found that the SW test has the following notable properties:

1. It is the most sensitive test when the data have a continuous, skewed, long-tailed distribution.
2. It is the most sensitive test when the data have a continuous, skewed, short-tailed distribution.
3. It is one of the four most sensitive tests when the data have a continuous, symmetrical, long-tailed distribution.
4. It is one of the two most sensitive tests when the data have a continuous, symmetrical short-tailed distribution.
5. It shares second place with another test when the true distribution is discrete.

Here sensitivity is measured by the power of the test, which is the probability of rejecting the null hypothesis of normality when in fact the sample data are nonnormal.

The SW test (Shapiro and Wilk, 1965) is based on an estimate of the squared slope of the regression line obtained when a random sample is plotted on normal probability paper. Under conditions of IID normality, this quantity is an estimate of the population variance multiplied by a constant. Nonnormality usually causes this quantity to be small relative to the corresponding sample variance. To determine the SW test statistic, W, for a sample size n, the following steps are required:

1. Compute $\underset{\sim}{m}' = (m_1, m_2, \cdots, m_n)$, the vector of expected values of standard normal order statistics for a sample of size n.
2. Compute $\underset{\sim}{V} = (v_{i,j})$, the corresponding n x n covariance matrix

for standard normal order statistics.

3. Compute $\underset{\sim}{Y}' = (Y_1, Y_2, \cdots, Y_n)$, the vector of ordered observations from the population to be tested. Let μ and σ^2 respectively denote the (unknown) mean and variance of this population.

4. Let $X_1 \leqslant X_2 \leqslant \cdots \leqslant X_n$ denote the corresponding ordered random sample from a standard normal distribution.

5. If the Y_i are normallv distributed, then we have the regression equation $Y_i = \mu + \sigma X_i$; and the weighted least-squares estimate of σ is

$$\hat{\sigma} = \frac{\underset{\sim}{m}' \underset{\sim}{V}^{-1} \underset{\sim}{Y}}{\underset{\sim}{m}' \underset{\sim}{V}^{-1} \underset{\sim}{m}} \quad . \tag{2.3.1}$$

With the definitions

$$R^2 = \underset{\sim}{m}' \underset{\sim}{V}^{-1} \underset{\sim}{m} , \tag{2.3.2}$$

$$C^2 = \underset{\sim}{m}' \underset{\sim}{V}^{-1} \underset{\sim}{V}^{-1} \underset{\sim}{m} , \tag{2.3.3}$$

the quantity

$$b = R^2 \hat{\sigma} / C \tag{2.3.4}$$

is, up to the normalizing constant C, the best linear unbiased estimate of the slope of a linear regression of the ordered observations, Y_i, on the expected values, m_i, of the standard normal order statistics.

6. The usual unbiased estimate of $(n-1)\,\sigma^2$ is

$$S^2 = \sum_{i=1}^{n} (Y_i - \bar{Y})^2 \tag{2.3.5}$$

7. The W test statistic is

$$W = \frac{b^2}{S^2} \qquad (2.3.6)$$

where (a) the distribution of W depends only on n, (b) the closer W is to unity (its maximal value), the more normal the data appear.

Since the $\underset{\sim}{m}'$ vector is known for a given n, Shapiro and Wilk were able to simplify the calculation of b bv tabulating values of a coefficient, a_{n-i+1}, which is used in their recommended test procedure.

1. Order the observations:

$$Y_1 \leqslant Y_2 \leqslant \cdots \leqslant Y_n$$

2. Compute

$$S^2 = \sum_{i=1}^{n} (Y_i - \bar{Y})^2$$

3. If n is even, set $j = n/2$; otherwise set $j = (n-1)/2$.

   Compute

$$b = \sum_{i=1}^{j} a_{n-i+1} (Y_{n-i+1} - Y_i)$$

   where values of a_{n-i+1} are given in Shapiro and Wilk (1965).

4. Compute $W = b^2/S^2$.

5. The 1, 2, 5, 10, 50, 90, 95, 98, and 99 per cent points of the distribution of W are given in Shapiro and Wilk (1965). Small values of W are significant, i.e. indicate non-normality.

   An impor..t improvement to this procedure was presented by

Shapiro and Francia (1972) when they formulated a similar test statistic, W', for sample sizes up to 400. Weisberg and Bingham (1975) published a computing formula for a test statistic, W', which they showed to be asymptotically equivalent to the two previous test statistics. This formula enables the calculation of the Shapiro and Wilk test statistic for any sample size.

2.4 Spectral Analysis of Simulation Data

Many authors have noted that in general, the data generated by computer simulation experiments are highly autocorrelated (Naylor et al. 1966; Law and Kelton 1982; Heidelberger and Welch 1981a). Yet classical statistical theory for estimating the variance of cumulative statistics like the sample mean is based on the assumption that the observations are independent and identically distributed. In many simulation experiments, the autocorrelation present in output series of interest causes classical statistical techniques to yield substantial underestimates of the variance of relevant cumulative statistics (Naylor, Wertz, and Wonnacott 1969). Use of such underestimates when calculating confidence intervals around the sample mean, for example, will cause unrealistically narrow intervals. One way to adequately account for autocorrelation is by batching the observations so that the resulting batch means are approximately normal and uncorrelated (and hence independent); see Law and Kelton (1982). Another method is to replicate the simulation experiment and compute the variance across replications; see Fishman (1978). The

regenerative method of analysis (Law and Kelton 1982) organizes the observations from a single run into cycles that are exactly IID; a steady-state performance estimator is then based on the ratio of relevant cycle measurements that have been averaged over all regenerative cycles observed during the run.

All of these methods for simulation analysis have serious drawbacks. Fishman and Kiviat (1967) have noted that in implementations of the method of batch means, procedures for determining the batch size "seem to have neither enough prior nor posterior justification in most cases to make a choice that is much more than arbitrary." Initialization bias is the main problem connected with the method of independent replications. In the case of regenerative analysis, regeneration frequencv is the issue: in most real-world models, the time intervals between successive regeneration epochs are much too large to allow an adequate number of regenerative cycles to be completed within a feasible run length. An alternative estimation procedure which avoids all of these difficulties can be based on spectral analysis (Brillinger 1975).

The spectral method used in this research to estimate the variance of the sample mean was developed by Heidelberger and Welch (1981a). By working with the periodogram of a simulation-generated time series, they converted the problem of dealing with the original autocorrelated series into the more tractable problem of fitting an appropriate curve to the uncorrelated observations that constitute the periodogram. Heidelberger and Welch assumed that the series (X_1, ...,

X_N) is a sample from a covariance stationary process with mean μ and lag-k autocovariance $\gamma_k (k = 0, \pm 1, \pm 2, \ldots)$ such that

$$\sum_{k=-\infty}^{\infty} |\gamma_k| < \infty. \tag{2.4.1}$$

This ensures the existence of the spectral density

$$p(f) \equiv \sum_{k=-\infty}^{\infty} \gamma_k \cos(2\pi fk). \tag{2.4.2}$$

As an estimator of μ, the sample mean $\bar{X}_N$ has variance

$$\mathrm{Var}(\bar{X}_N) = N^{-1} \sum_{k=-(N-1)}^{N-1} (1-|k|/N)\gamma_k; \tag{2.4.3}$$

and in view of (2.4.1) and (2.4.2), we have

$$\lim_{N \to \infty} N*\mathrm{Var}(\bar{X}_N) = \sum_{K=-\infty}^{\infty} \gamma_k = p(0). \tag{2.4.4}$$

The spectral method of simulation analysis uses the large-sample approximation

$$\mathrm{Var}(\bar{X}_N) = p(0)/N, \tag{2.4.5}$$

so that it is only necessarv to estimate the spectral density at zero frequency.

Heidelberger and Welch noted that the usual methods of spectral estimation (Jenkins and Watts 1968, Bloomfield 1976) are not appropriate for estimating the spectrum at zero frequency. Such methods use a spectral window $H(f)$ so that the estimator $\hat{p}(f^*)$ at a particular frequency f^* has for its expected value a weighted average of the spectrum $p(f)$ in a neighborhood of f^*:

$$E[\hat{p}(f^*)] = \int_{-1/2}^{1/2} H(f - f^*)\ p(f)\ df \qquad (2.4.6)$$

Now the spectrum has a local optimum at $f^* = 0$; and in many queueing simulations the spectrum is sharplv peaked at zero frequencv. Equation (2.4.6) shows that in such a situation, a classical estimator $\hat{p}(0)$ has a large negative bias. To reduce the bias of $\hat{p}(0)$, we must use a narrow window. Unfortunatelv the variance of any estimate of $\hat{p}(0)$ increases as the width of the window decreases. Thus classical methods of spectral estimation yield either a highly variable estimator of p(0) which is approximately unbiased, or a stable estimator which is strongly biased. Duket and Pritsker (1978) investigated these problems experimentally for the queue length process in a single-server queue with exponential interarrival and service times.

Heidelberger and Welch developed the following procedure to overcome the problems of classical spectral estimation:

1. Calculate the periodogram

$$I(n/N) = N^{-1} \left| \sum_{j=1}^{N} X_j * \exp[-2\ \pi i(j-1)(n-1)/N] \right|^2 \qquad (2.4.7)$$

for $n = 1, \ldots, 2K$, where $i = (-1)^{1/2}$ and $K = N/4$.

2. Calculate the logarithm of the "smoothed" periodogram:

$$f_n = (4n-1)/(2N), \qquad (2.4.8)$$

and

$$J(f_n) = \log[\ (\ I[(2n-1)/N] + I[2n/N]\ )/2\ ] \qquad (2.4.9)$$

for $n = 1, \ldots, K$.

3. Using ordinary least-squares, fit a polynomial of degree d

$$g(f_n) = \sum_{k=0}^{d} a_k \, f_n^k \tag{2.4.10}$$

to the function $J(f_n) + 0.270$ for $n=1, \ldots, K$.

4. Using the least-squares estimate $\hat{a}_0$ and the design matrix

$$\underset{\sim}{X} = \begin{bmatrix} 1 & f_1 & \cdots & f_1^d \\ \cdot & & \cdots & \\ 1 & f_K & \cdots & f_K^d \end{bmatrix}, \tag{2.4.11}$$

associated with the regression (2.4.10), compute the quantity

$$C_1(K,d) = \exp\left(-0.3225[(X'X)^{-1}]_{11}\right) \tag{2.4.12}$$

in order to obtain the approximatelv unbiased estimator

$$\hat{p}(0) = C_1(K,d)*\exp(\hat{a}_0) \tag{2.4.13}$$

for the spectrum at *zero frequency*.

The regression (step 3) provides the stability of averaging over a number of periodogram values, and the flexibility of the family of fitted functions avoids the bias associated with the spectral window. The advantage of operating in the frequencv domain versus the time domain is shown bv the fact that

$$\mathrm{Cov}[J(f_m), J(f_n)] \cong 0, \qquad m \neq n. \tag{2.4.14}$$

Batching the periodogram into batches of size 2 and taking the logarithm of the batch means (see equation (2.4.9)) is intended to stabilize the variance of the periodogram so that ordinary least-squares can be used to fit the polynomial in step 3. In a subsequent article, Heidelberger and Welch (1981b) found that using a polynomial, of degree $d = 2$ in (2.4.10) provides the best trade-off between the bias and the variance of $\hat{p}(0)$.

CHAPTER III

DEVELOPMENT OF THE MULTIPLE RANKING PROCEDURE

3.1 Introduction

To develop a multiple ranking procedure that can handle simulation-generated time series characterized by initialization bias, unknown autocovariance structures, and nonnormality, appropriate algorithms were developed for each aspect of the problem. In particular, separate support routines were designed to perform the following operations on such data sets:

1. Induce normality by adequate batching
2. Eliminate initialization bias with a cusum test
3. Compute the final sample size with an estimate of the spectrum at zero frequency replacing the sample variance in the Dudewicz-Dalal formula.

The organization of this chapter reflects this three-way division of the overall problem. The section devoted to each subproblem includes the appropriate theoretical development, a program description, and a summary of the procedures used in the verification and validation of the algorithm. A final section describes the integration of these routines into the complete MRP adapted to discrete-event simulation.

3.2 Normality Test Procedure

3.2.1 Development

A stand-alone computer program written by De Branges (1974) was used as the foundation for building a support routine to perform the Shapiro-Wilk normality test at a user-specified level of significance. The program, WILK, determines the batch size, NSIZE, required to induce an acceptable degree of convergence to normality in the data produced by a discrete-event simulation. The original program by De Branges required extensive modification to eliminate extraneous calculations and to allow the inclusion of repeated testing on increasingly larger batch sizes.

WILK uses an array containing data from up to 16 replications of the simulation to be examined, where each independent replication generates up to 100 data points. The replication count, r, and the sample size, n_o, within each replication are selected by the simulator prior to execution. The series

$$(X_{ij} : \ i = 1, \ldots, r \text{ and } j = 1, \ldots, n_o)$$

is first tested for normality at the $\alpha = 0.1$ level of significance by computing batch means across batches of size NSIZE:

$$\bar{X}_i(\text{NSIZE}) = (\text{NSIZE})^{-1} \sum_{j=1}^{\text{NSIZE}} X_{ij}, \ i = 1, \ldots, r, \tag{3.1.1}$$

starting with NSIZE = 1. The composite hypothesis tested is:

$$H_o(\text{NSIZE}): \{\bar{X}_i(\text{NSIZE}) \ \ 1 \leqslant i \leqslant r\} \sim \text{IID } N(\mu, \sigma^2) \tag{3.1.2}$$

$$\text{for some } \mu \text{ and } \sigma$$

If H_o(NSIZE) is rejected, the value of NSIZE is increased by one and

(3.1.2) is tested again; otherwise the value of NSIZE is returned for use by other parts of the analysis program. If the test for normality is rejected when NSIZE equals NMAX (NMAX = 3 by default), WILK reports back to the user that convergence to normality is not achieved. The user has the option to either increase the number of replications or the run length. Figure 3.1 provides a flowchart of WILK. A listing of the program, WILK, is presented in Appendix A.

3.2.2 Verification and Validation

The program to perform the Shapiro-Wilk test for normality was used to test three data sets that had been previously tested by the original De Branges program. In each case, with NSIZE set equal to 1, identical results were obtained.

Several highly correlated data sets were also generated and passed to WILK in order to test the operation of the batching procedure. A variety of stationary autoregressive-moving average processes were generated by the support routine ARMAPQ (Hoffman 1982) for use in this phase of the program verification. Table 3.1 summarizes the results produced by WILK. The same test sequences, both original data and batched data, were also tested by the original De Branges program and identical results were obtained

3.3 Initialization Bias Test Procedure

3.3.1 Development

The initialization bias (IB) detection routine is designed to

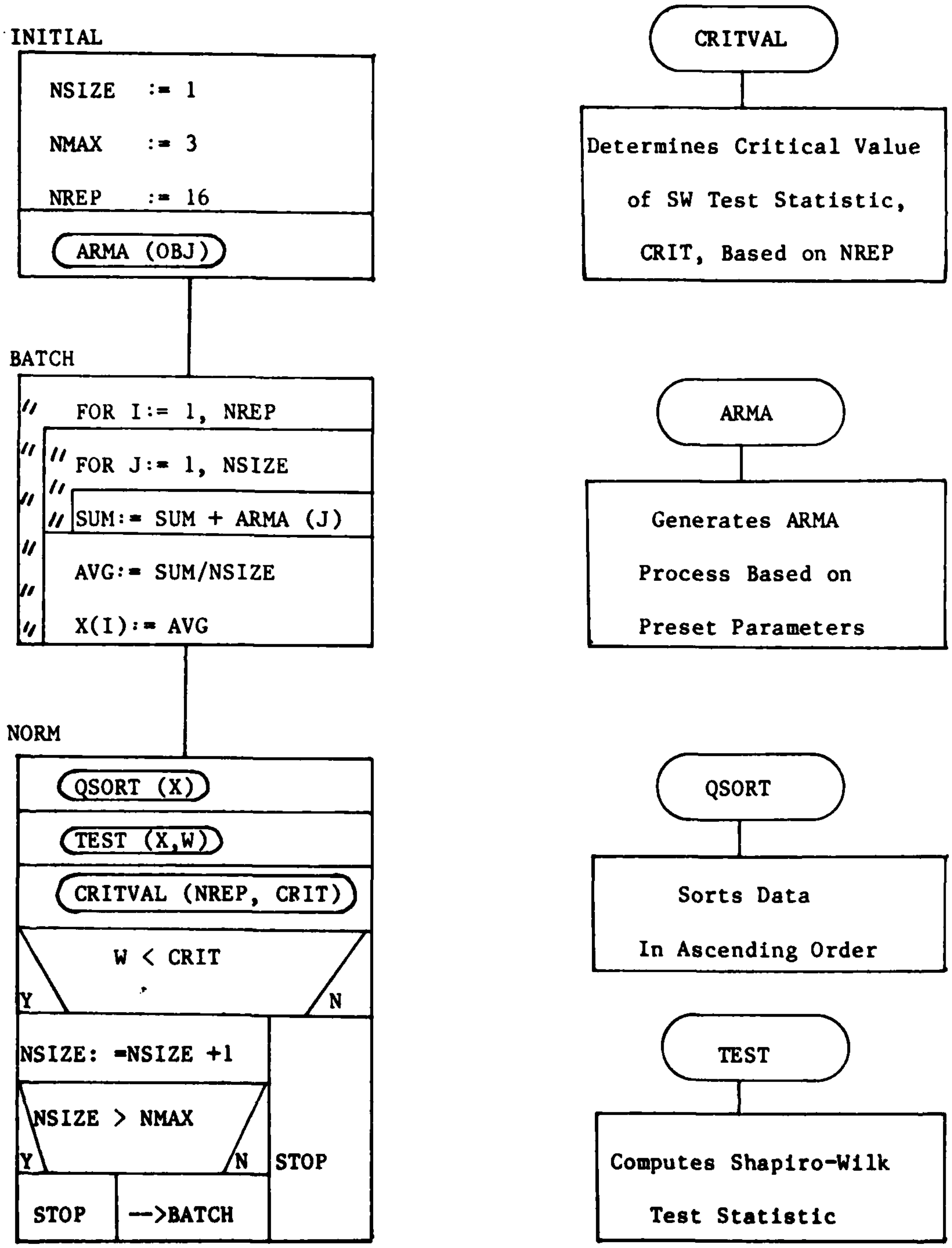


Figure 3.1 Program WILK Flowchart

Table 3.1 Results of testing subprogram WILK with ARMA (p,q) processes.

AR PARAMETERS	MA PARAMETERS	MEAN μ	WHITE NOISE VARIANCE σ^2	FINAL NSIZE
- 0.35, 0.25	•	100	400	1
0.35, 0.25	•	100	400	1
0.65, 0.25	•	100	400	2
0.65, 0.25	•	100	25	1
0.75, 0.75	•	50	400	1
- 0.8	•	50	100	1
0.85	•	50	100	1
•	0.35	50	100	1
•	0.6	10	25	1
•	0.4, 0.4	10	25	2
•	0.25, - 0.75	100	100	2
•	- 0.6, 0.4	25	49	1
0.4	0.4	25	49	2
- 0.2	0.8	100	100	1

test the output sequence from a discrete-event simulation after the sequence has been batched to achieve normality using program WILK. It is the responsibility of the user to specify the level of Type I error appropriate for his needs.

As discussed in the literature review, the most appealing method is Schruben's one-sided test (2.2.12) for negative IB that does not require an estimate of the process variance parameter (2.2.3). Its sensitivity and its independence of the underlying covariance structure make this test substantially more effective than any other available technique. Once this procedure had been coded, an experiment was performed with six data sets having each of the following characteristics:

1. Negative bias
2. Positive bias
3. A damped oscillation between negative and positive bias
4. Negative bias not reaching stationarity
5. Positive bias not reaching stationarity
6. No initial bias.

Figure 3.2 shows the graphs of each data set. The program includes a routine to identify positive bias and to account for it by reversing the sign of all observations.

The program performed well for data sets 1, 2, 4, 5, and 6 in that each occurrence of initial bias was detected and an intuitively reasonable truncation point was specified. However, the program failed when data set 3 was tested. The problem was found to be that

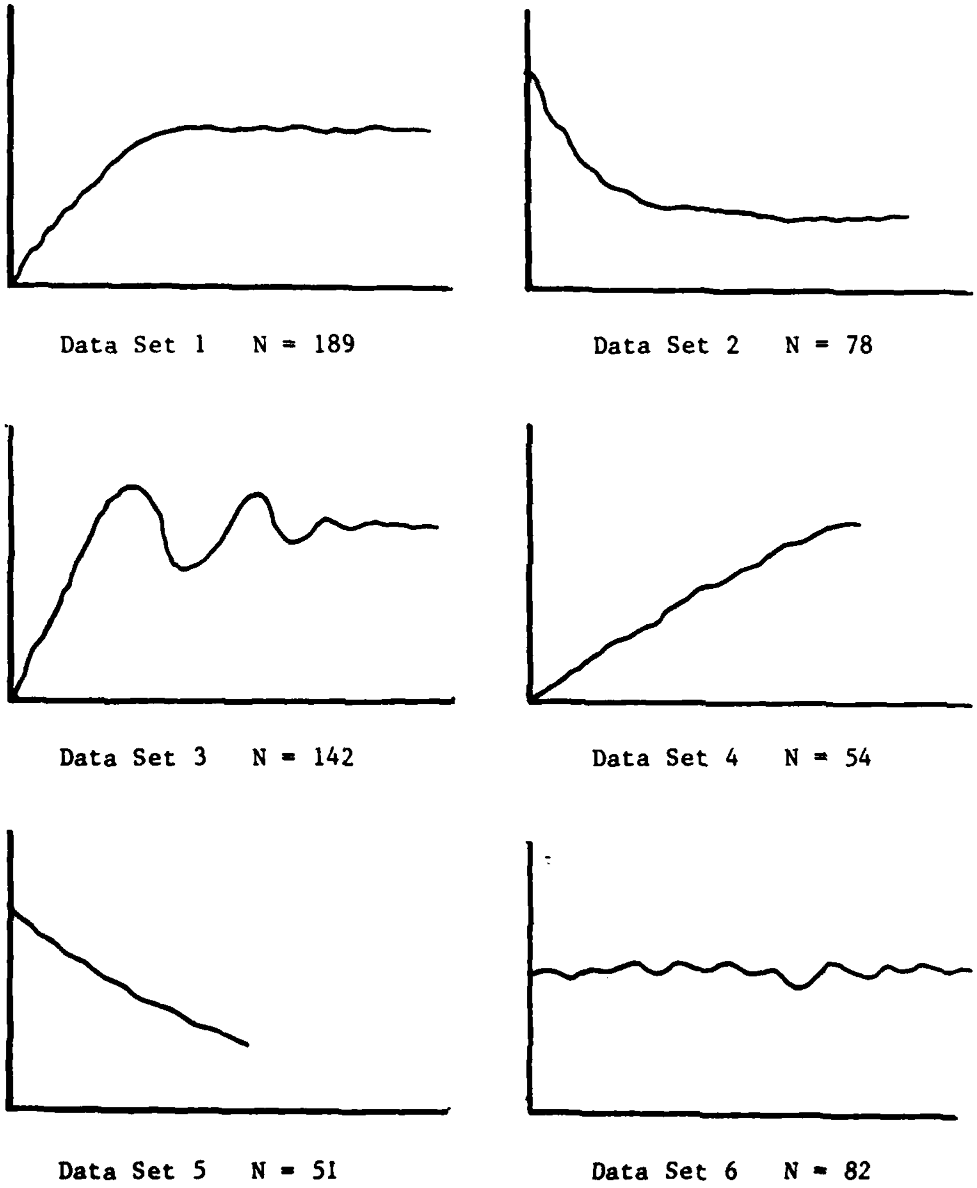


Figure 3.2 Data sets used to validate the initialization bias test.

the ratio $\hat{g}_f/\hat{g}_\ell$ of maximum squared cusums respectively computed from the first half and last halves of the data fails to detect significant bias when the two optima occur on opposite sides of the mean. Because the test procedure involves squaring the maximum cusum values, the signs of these deviations from the mean are lost. It is unclear how to modify the test statistic $\hat{g}_f/\hat{g}_\ell$ to handle this type of transient behavior.

As an alternative to the test statistic $\hat{g}_f/\hat{g}_\ell$, we coded Schruben's other one-sided statistic $\hat{g}$ defined by equation (2.2.11). The parameter σ^2 for the batch mean process defined by (2.2.3) is estimated bv computing the sample variance, V_s^2, of the "safe" batch means using equation (2.2.5). The lag-one correlation between the "safe" batch means

$$\hat{\rho}_1 = V_s^{-2} *(s-1)^{-1} \sum_{i=1}^{s-1} (\bar{X}_i - \bar{\bar{X}}_s)(\bar{X}_{i+1} - \bar{\bar{X}}_s) \qquad (3.3.1)$$

is then used to yield the final parameter estimate

$$\hat{\sigma}^2 = V_s^2 *[(1 + \hat{\rho}_1)/(1 - \hat{\rho}_1)] \qquad (3.3.2)$$

for the cusum test.

The motivation for (3.3.2) closelv parallels the analysis given by Dudewicz and Zaino (1977): in effect we are modeling the "safe" batch means as an autoregressive process of order one so that a simple formula for σ^2 can be applied. Only the first-order autoregressive effect is considered because it is equally important to detect actual initialization bias (i.e., to avoid Type II error) as it is important to avoid falsely detecting nonexistent bias (i.e., to

avoid Type I error). Therefore, we judged that it is better to underestimate σ^2 by using only lag 1 effects in (3.3.2) than to possibly overestimate σ^2 using higher-order effects. Completely ignoring these effects was considered to be too gross an underestimation. This same reasoning forms the basis for ignoring any negative autocorrelation.

The modified initialization bias test procedure based on (2.2.11) and (3.3.2) was applied to the six data sets depicted in Figure 3.2. In each case, the modified test correctly identified the presence of bias.

In correcting for the effects of initialization bias, the next step is to determine a suitable truncation point. This is accomplished by successively deleting blocks equal to 10% of the original data set until the remaining series finally yields a nonsignificant value for the test statistic. As in the experiments reported by Heidelberger and Welch (1982), the truncation procedure is stopped if 50% of the original data set has been truncated and the bias effects have still not been eliminated. Use of this limit is reinforced by the fact that the 50% point is used as the point where the "safe" data starts.

During the testing of data set 3 (Figure 3.2), disturbing results were obtained. Specifically, the test forced the leading 10% of the data to be truncated; but then a test on the remaining data points failed to produce a significant result. Figure 3.3 reveals marked transient behavior in the remaining time series. This casts

some doubt on the adequacy of <u>any</u> one-sided test for initialization bias in the presence of nonmonotonic transients.

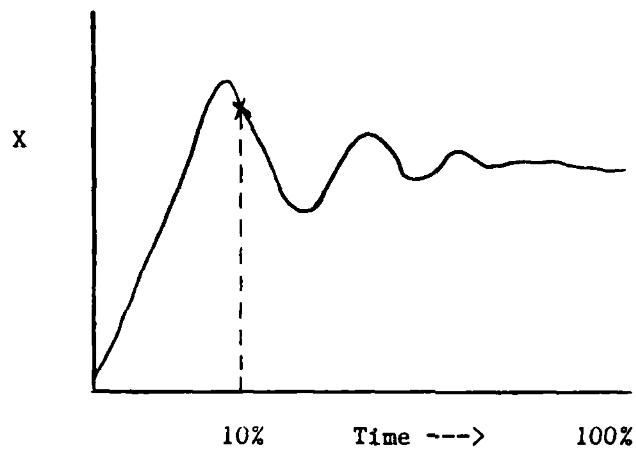


Figure 3.3 Truncation point determined by one-sided cusum test.

The intermediate results generated during the test were checked by hand, and a serious inadequacy was found. Namely, the procedure only checks for deviations from the mean in one direction. This fact clearly shows that Schruben's procedure only applies to monotonically decreasing transients. To accommodate the possibility of nonmonotonic transients, it was necessary to develop a two-sided procedure. Such a procedure will be less powerful than the corresponding one-sided test when the transient mean function lies entirely on one side of the steady-state mean, but a two-sided test is necessarv to handle the transients that frequently occur in queueing simulations.

Replacing the minimum cusum value in (2.2.11) with the maximum

absolute cusum,

$$S^{**} = \max \{ \, | \hat{S}_j | : 1 \leqslant j \leqslant b \, \} \qquad (3.3.3)$$

$$K^{**} = \min \{ j : | \hat{S}_j | = S^{**} \} \qquad (3.3.4)$$

yields a variant of Schruben's two-sided test procedure that avoids the problems encountered in the previous techniques. This was corroborated when the two-sided test was coded and applied to the 6 data sets shown in Figure 3.2. Using the automatic truncation routine previously described, we obtained favorable results. The troublesome properties exhibited by data set 3 were identified, and a reasonable truncation point was reported. Once a robust procedure for initial bias detection was found that would accept a complete range of possible inputs, it was decided to see if any of the desirable features of the previously rejected approaches could be incorporated into the final product.

Since the purpose of the test procedure is to identify and eliminate initial bias, we considered standardizing the two-sided test statistic so that it has the same asymptotic distribution as the absolute maximum of a Brownian bridge process:

$$B^{**} = S^{**}/(\hat{\sigma} [t^{**}(1-t^{**})]^{1/2}) \qquad (3.3.5)$$

where

$$t^{**} = K^{**}/b. \qquad (3.3.6)$$

Unfortunately the joint distribution of t^{**} and B^{**} is not known for a Brownian bridge process. To take advantage of the distribution theory underlying Schruben's one-sided test statistic (2.2.11), the following two-sided procedure was based on 2 one-sided tests and a Bonferroni

inequality:

1. Find S^+ and S^-, the maximum and minimum values of the cusum

$$S_k = b^{-1/2} \sum_{j=1}^{k} (\bar{X}_j - \bar{\bar{X}}_i), \; k = 1, \ldots, b, \qquad (3.3.7)$$

   together with the corresponding indices K^+, K^- where the maximum and minimum respectively occur.

2. If S^+ or S^- equals zero (indicating cusums of onlv one sign), set

$$S = \max \{ S^-, S^+ \}, \qquad (3.3.8)$$

   and proceed to the previously given one-sided test (2.2.11). Otherwise go to step 3.

3. Compute $\hat{\sigma}^2$ by equations (3.3.1) and (3.3.2).

4. Set ν, the number of degrees of freedom, equal to (b-1), where b is the number of batches.

5. Set $t^+ = K^+/b$ and $t^- = K^-/b$.

6. Compute

$$g^+ = (S^+)^2 / [3 \quad \hat{\sigma}^2 \; t^+(1-t^+)] \qquad (3.3.9)$$

$$g^- = (S^-)^2 / [3 \quad \hat{\sigma}^2 \; t^-(1-t^-)] \qquad (3.3.10)$$

7. Use the IMSL routine MDFDRE to determine the significance probabilities.

$$\alpha^+ = \Pr \{ F_{3,\nu} > g^+ \} \qquad (3.3.11)$$

$$\alpha^- = \Pr \{ F_{3,\nu} > g^- \} \qquad (3.3.12)$$

8. Reject a hypothesis of no bias at the α level of significance if:

$$\min(\alpha^-, \alpha^+) < \alpha/2. \qquad (3.3.13)$$

In this two-sided test, the maximum prespecified level of Type I error, α , is maintained by performing 2 complimentary one-sided tests each at significance level $\alpha/2$. This modification will ultimately result in a somewhat larger sample size than would be required by a test with size exactly equal to α (Bowker and Lieberman 1972). The validity of rejecting the null hypothesis of no bias

$$H_o: \ E[\bar{X}_i] = \mu, \ i = 1, \ldots, b \qquad (3.3.14)$$

when min (α^-, α^+) < $\alpha/2$ is based on the following conditional Bonferroni inequality:

$$\begin{aligned} \Pr \{ \text{Accept } H_o \mid H_o \} &= \Pr \{ \min(\alpha^-, \alpha^+) \geqslant \alpha/2 \mid H_o \} \\ &= \Pr \{ \alpha^- \geqslant \alpha/2 \ \underline{\text{and}} \ \alpha^+ \geqslant \alpha/2 \mid H_o \} \\ &\geqslant 1 - \Pr \{ \alpha^- \geqslant \alpha/2 \mid H_o \} \\ &\qquad - \Pr \{ \alpha^+ \geqslant \alpha/2 \mid H_o \} \\ &= 1 - \alpha \qquad (3.3.15) \end{aligned}$$

As mentioned earlier, the data sequence ($\bar{X}_i$: i = 1, ..., b) tested by the IB detection routine was formed by taking the mean value of NSIZE original data points i.e.

$$\bar{X}_i = (\text{NSIZE})^{-1} \sum_{k=(i-1)*\text{NSIZE}+1}^{i*\text{NSIZE}} X_k, \ i = 1, \ldots, b = n_o/\text{NSIZE} \qquad (3.3.16)$$

This batching was found to have no effect on identification of initialization bias.

Initially the truncation point was chosen arbitrarily at the 0.1 * b data point (b = number of batches). Since the IB test procedure specifically identifies an epoch beyond which the null hypothesis is accepted, the truncation of the data at that particular

point is more reasonable than deleting 10% of the data. If a two-sided test is necessary, then K^+ is used for a truncation point when $\alpha^+ < \alpha/2$, and K^- is used when $\alpha^- < \alpha/2$. When both α^+ and α^- are less than $\alpha/2$, the larger of K^+ and K^- determines the truncation point.

While working on subsequent stages of the dissertation, we discovered a superior method for estimating the variance parameter σ^2 of a correlated data sequence. (See section 3.4.3). Note that σ^2 is just the spectrum at zero frequency p(0) for the batch mean process (3.3.16). Thus the spectral analysis routine, WELCH, is used in place of equations (3.3.1) and (3.3.2) in the final version of the initialization bias detection routine, IBZERO. A listing of IBZERO is included in Appendix B. A flowchart of IBZERO is shown in Figure 3.4.

3.3.2 Verification and Validation

The data used during the development of the initialization bias detection program, IBTEST, were also used to test the final version of the program, IBZERO. As noted previously, IBZERO estimates the process variance parameter, σ^2, by estimating the spectral density at zero frequency. Table 3.2 compares the results of IBTEST and IBZERO. In the cases where 50% of the data was truncated and the remaining series still showed the effects of initial bias (data sets 4 and 5), the user was given the warning message that a larger sample size would be required to eliminate initial bias effects.

Additional data sets were generated using the

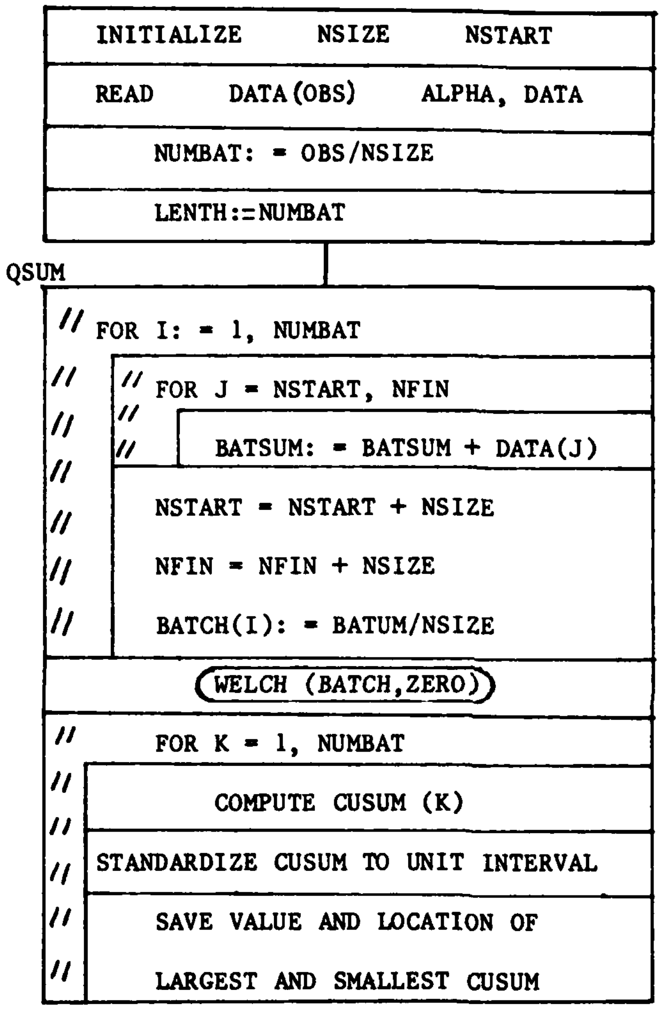


Figure 3.4 Flowchart of initialization bias test procedure IBZERO

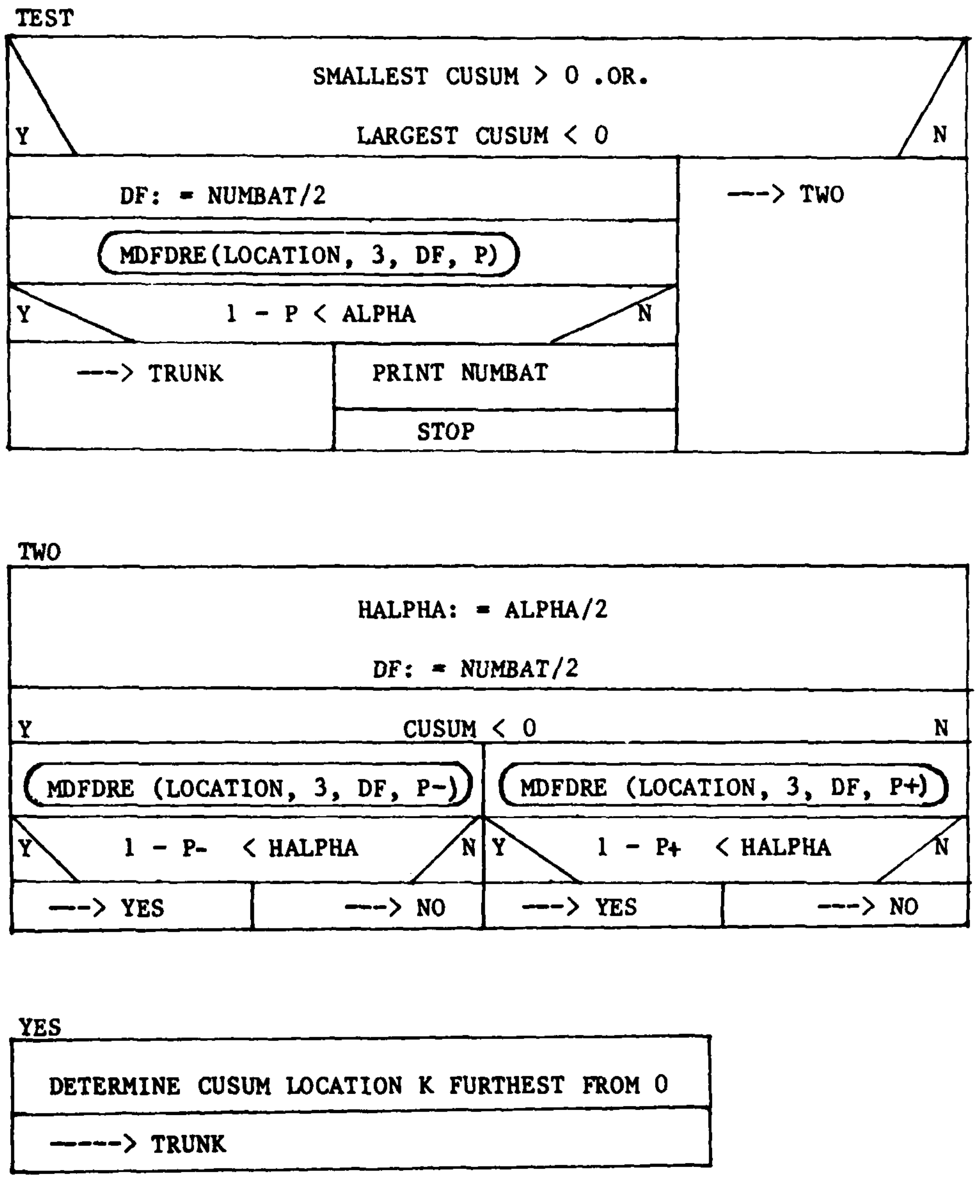


Figure 3.4 (continued)

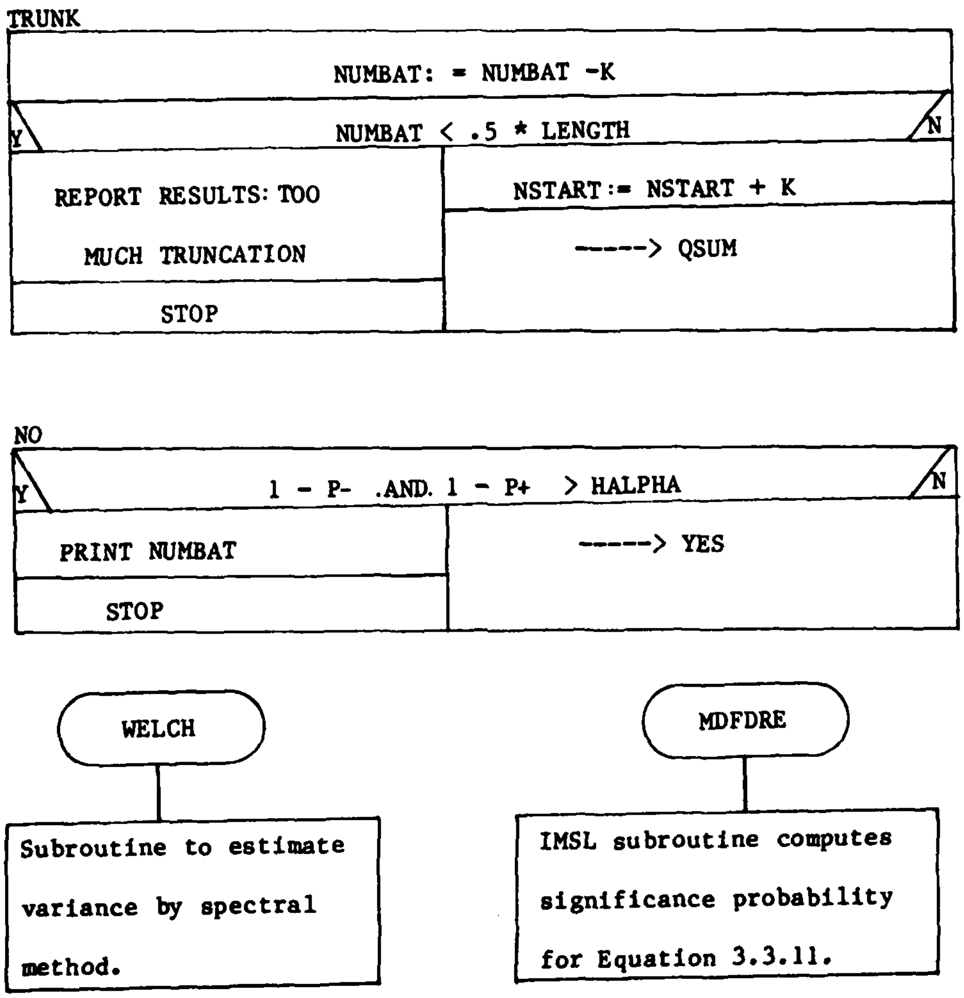


Figure 3.4 (continued)

Table 3.2 Performance comparison for the data-truncaton procedures IBTEST and IBZERO.

DATA SET	PROCEDURE	TRUNCATION POINT	ESTIMATE OF μ	ESTIMATE OF σ^2
1	IBTEST	26	11.05	0.32
1	IBZERO	26	11.05	0.31
2	IBTEST	19	35.39	2.12
2	IBZERO	19	35.39	1.94
3	IBTEST	49	10.18	0.036
3	IBZERO	40	10.12	0.019
4	IBTEST	DID NOT REACH		20.4
4	IBZERO	STATIONARITY		4.7
5	IBTEST	DID NOT REACH		918
5	IBZERO	STATIONARITY		820

autoregressive-moving average process generator ARMAPQ. Each series was generated with a zero mean; then various types of IB were superimposed by adding a transient mean function. Each sequence was finally tested by IBZERO. To show graphically the transient behavior of the input series as well as the cusum values used during the test, a plot routine was written and incorporated into IBZERO.

Figures 3.5 thru 3.8 show the results of testing such a sequence. The test series is the same one used to test program WILK. Figure 3.5 shows the results of the interactive terminal session in which IBZERO was used to determine a truncation point for the series. Figure 3.6 is a plot of the batch means. For this test the batch size, NSIZE, was set equal to three. Therefore the requested test of 120 original data points resulted in a test of 40 batch means. Figure 3.7 shows the plot of the cusum produced by the 40 batch means. This figure displays the early peak that is characteristic of negative initial bias.

Based on the peak of the cusum occuring at batch number 6, the leading 18 observations of the original series were deleted (note that 6*NSIZE = 18 when NSIZE = 3). A subsequent test on the remaining data, Figure 3.8, showed that there was no pronounced early peak; and the hypothesis of no initial bias was accepted. It should be noted that the graphs in Figures 3.7 and 3.8 have different scales. Before the IB was eliminated, the cusum varied from 2.0 to -2.2 (see Figure 3.7). After deletion of the transient observations, the cusum fell in the range 0.6, -0.2 (see Figure 3.8). Since a fluctuating initial

ENTER THE NUMBER OF DATA POINTS

< 120 >

ALPHA =

< 0.1 >

DO YOU WANT TO RERUN WITH A DIFFERENT LEVEL OF SIGNIFICANCE?

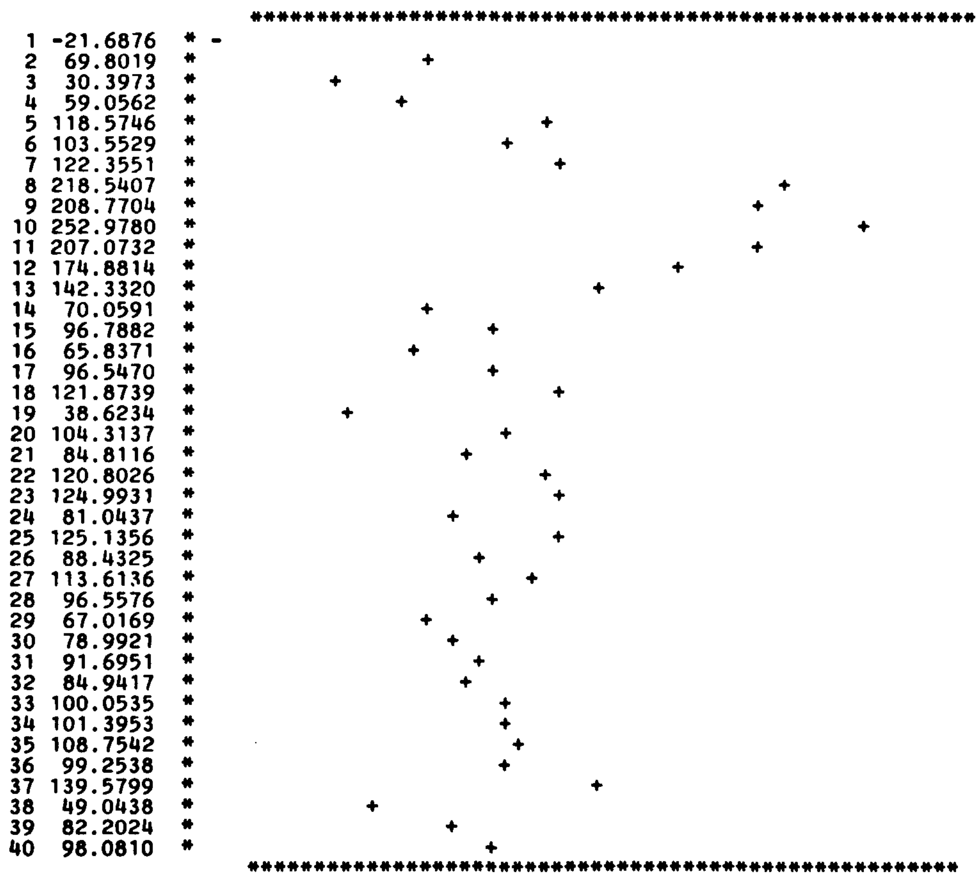
ENTER 0 FOR NO

ENTER 1 FOR YES

< 0 >

TEST COMPLETE

Figure 3.5 Interactive session for testing transient AR(1) series tested by procedure IBZERO.



PLOT OF BATCH MEANS

Figure 3.6 Transient AR(1) series tested by IBZERO.

1	.8912
2	1.1410
3	1.6671
4	1.9922
5	1.9000
6	1.9132
7	1.7945
8	1.0014
9	.2768
10	-.7577
11	-1.4704
12	-1.9574
13	-2.2162
14	-1.9682
15	-1.9076
16	-1.6300
17	-1.5678
18	-1.6831
19	-1.2147
20	-1.2069
21	-1.0624
22	-1.1702
23	-1.3074
24	-1.1364
25	-1.2746
26	-1.1554
27	-1.2128
28	-1.1506
29	-.8813
30	-.6960
31	-.5997
32	-.4561
33	-.4184
34	-.3902
35	-.4135
36	-.3702
37	-.6097
38	-.2143
39	-.0515
40	0

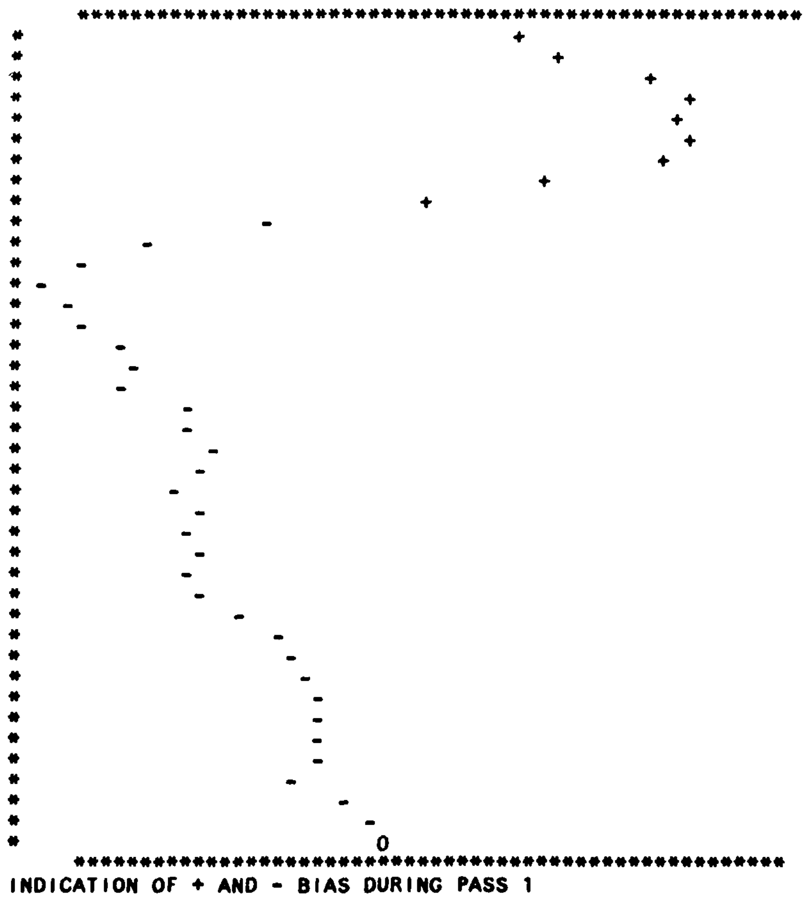
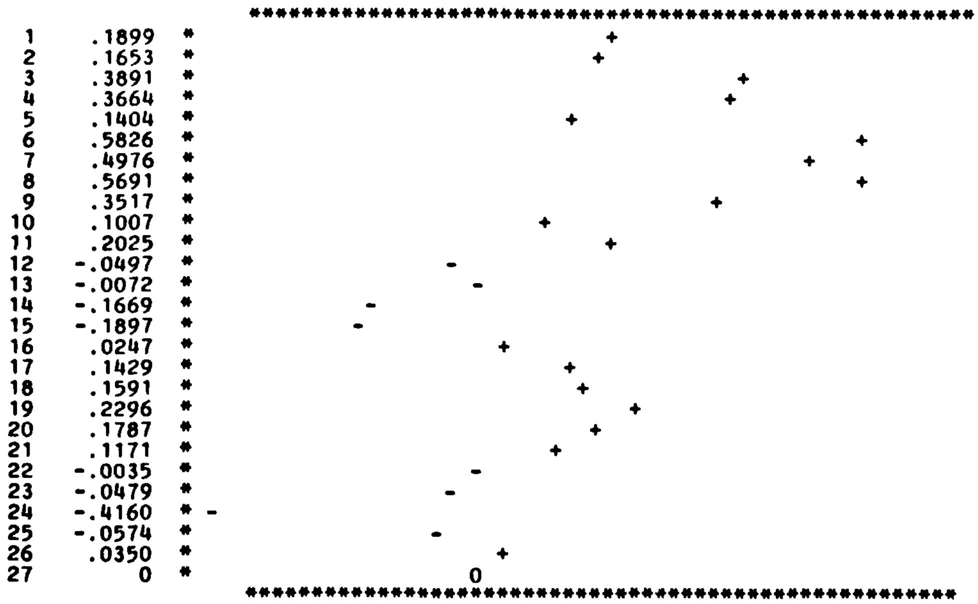


Figure 3.7 CUSUM for transient AR(1) series tested by IBZERO.



INDICATION OF + AND - BIAS DURING PASS 3

HYPOTHESIS OF NO INIIALIZATION BIAS IS NOT REJECTED AT SPECIFIED LEVEL OF SIGNIFICANCE .1000 WITH A TRUNCATION POINT OF 13

MEAN OF RAW DATA = 105.4267

MEAN OF THE TRUNCATED DATA = 98.7201

Figure 3.8 CUSUM for AR(1) series truncated by IBZERO.

bias was found in this particular example, the more powerful one-sided IB test could not have been used.

3.4 Multiple Ranking Procedure for Correlated Processes

3.4.1 Extension of Dudewicz-Dalal Procedure

As shown in Section 2.1.2, previous attempts to develop multiple ranking procedures for correlated processes have been restricted in scope to simple autoregressive processes of order one. Unfortunately, a large number of simulation-generated processes which can be modeled by time series do not fit this convenient AR(1) form. To characterize such a process $(Z_t: \; t = 1, 2, \ldots)$ adequately may require the introduction of:

1. Higher-order autoregressive terms in, say, an AR(p) model

$$Z_t = \mu_Z + \phi_1(Z_{t-1} - \mu_Z) + \ldots + \phi_p(Z_{t-p} - \mu_Z)\, a_t \qquad (3.4.1)$$

2. Higher-order moving average terms in, say, an MA(q) model

$$Z_t = \mu_Z + a_t - \theta_1 a_{t-1} - \ldots - \theta_q a_{t-q} \qquad (3.4.2)$$

3. Both autoregressive and moving average terms in, say, an ARMA(p,q) model

$$Z_t = \mu_Z + \phi_1(Z_{t-1} - \mu_Z) + \ldots + \phi_p(Z_{t-p} - \mu_Z) + a_t - \theta_1 a_{t-1} \cdots - \theta_q a_{t-q}. \qquad (3.4.3)$$

If the MRP of Dudewicz and Zaino (1977) for AR(1) processes can be generalized to include ARMA(p,q) processes, then a large class of discrete-event simulations can then be tested to find the "best" alternative. However, it is known that many of the covariance

stationary processes generated by simulation models in steady-state operation do not have a finite-order representation in any of the forms (3.4.1), (3.4.2), or (3.4.3) (Fishman, 1973, p. 185). In this section we extend the Dudewicz-Dalal multiple ranking procedure to ARMA(p,q) processes and then to general covariance stationary processes.

In the case of a stationary AR(1) process with white noise variance σ_a^2 (representation (3.4.1) with $p = 1$), the key to the Dudewicz-Zaino procedure is the observation that the sample mean

$$\bar{Z}_n = n^{-1} \sum_{t=1}^{n} Z_t \qquad (3.4.4)$$

has variance

$$\mathrm{Var}(\bar{Z}_n) = n^{-1}[\sigma_a^2/(1 - \phi_1^2)](1 + \phi_1)/(1 - \phi_1) + O(n^{-2}). \qquad (3.4.5)$$

In terms of $\sigma_Z^2 = \sigma_a^2/(1 - \phi_1^2)$, we have

$$\gamma_Z = \lim_{n \longrightarrow \infty} n \cdot \mathrm{Var}(\bar{Z}_n) = \sigma_Z^2 \cdot (1 + \phi_1)/(1 - \phi_1). \qquad (3.4.6)$$

In effect Dudewicz and Zaino used the large-sample approximation

$$\mathrm{Var}(\bar{Z}_n) \cong \gamma_Z/n = (\sigma_Z^2/n) \cdot (1 + \phi_1)/(1 - \phi_1) \qquad (3.4.7)$$

to derive their sample size formula for ranking K alternative AR(1) processes with indifference zone width Δ^* and probability of correct selection P^*:

$$N_2 = h^2(P^*, K) \cdot \gamma_Z/(\Delta^*)^2 \qquad (3.4.8)$$

$$= [h^2(P^*, K) \cdot \sigma_Z^2/(\Delta^*)^2] \cdot (1 + \phi_1)/(1 - \phi_1). \qquad (3.4.9)$$

The term $(1 + \phi_1)/(1 - \phi_1)$ in (3.4.9) is called the run length "inflation factor." Since the data are correlated, each observation

yields less new information about the process mean than if the data were independent. Therefore, the run length must be increased to account for the "loss" of information caused by the correlation. As expected, if the correlation between observations increases, then the required run length will also increase.

Now a stationary ARMA(p,q) process can be represented in the form

$$Z_t = \sum_{j=0}^{\infty} \Psi_j a_{t-j} \qquad (\Psi_o = 1), \tag{3.4.10}$$

where the function

$$\Psi(\xi) = \sum_{j=0}^{\infty} \Psi_j \xi^j \tag{3.4.11}$$

of the complex variable ξ is analytic in the open disk $|\xi| \leqslant 1 + \delta$ for some $\delta > 0$. Since the white noise process (a_t) is uncorrelated, it is straightforward to show that

$$\gamma_k = \mathrm{Cov}(Z_t, Z_{t+k}) = \sigma_a^2 \sum_{j=0}^{\infty} \Psi_j \Psi_{j+k} \tag{3.4.12}$$

for $k \geqslant 0$. It follows that

$$\gamma_Z = \sum_{k=-\infty}^{+\infty} \gamma_k = \sigma_a^2 \cdot \Psi^2(1) . \tag{3.4.13}$$

If we define the polynomials

$$\Phi(\xi) = 1 - \phi_1 \xi^1 - \ldots - \phi_p \xi^p , \tag{3.4.14}$$

$$\theta(\xi) = 1 - \theta_1 \xi^1 - \ldots - \theta_q \xi^q , \tag{3.4.15}$$

then $\Phi(\xi)$ has no roots in the disk $|\xi| < 1 + \delta$, and we have (Box and Jenkins, 1976, p. 53)

$$\Psi(\xi) = \theta(\xi)/\Phi(\xi) \qquad \text{for} \quad |\xi| < 1 + \delta \tag{3.4.16}$$

Combining (3.4.13) and (3.4.16), we finally obtain a result analogous to (3.4.6)

$$\gamma_Z = \sigma_a^2 \cdot (1 - \theta_1 - \cdots - \theta_q)^2/(1 - \phi_1 - \cdots - \phi_p)^2 \,. \tag{3.4.17}$$

Thus the analogue of the sample-size formula (3.4.8) for a stationary ARMA(p,q) process is

$$N_2 = h^2(P^*, K)\cdot(\Delta^*)^{-2} \cdot \sigma_a^2 \cdot \frac{(1 - \theta_1 - \cdots - \theta_q)^2}{(1 - \phi_1 - \cdots - \phi_p)^2} \,. \tag{3.4.18}$$

There are two major drawbacks to the use of a multiple ranking procedure based on (3.4.18). As previously mentioned, the first and most basic problem is that some simulation output processes do not have a finite-order ARMA representation. The second problem is that (3.4.18) requires the implementation of subroutines to perform automatic ARMA model identification and parameter estimation. In an extensive experimental investigation of the use of automatic ARMA modeling algorithms in queueing simulation, Schriber and Andrews (1982) reported poor coverage for the associated confidence interval estimators. Although (3.4.18) sheds new light on the central issues involved in ranking correlated processes, a more flexible and robust approach is required for application to discrete-event simulation.

Spectral analysis provides an alternative means of extending the basic work of Dudewicz and Zaino. If the covariance stationary process (Z_t) has the spectral density function

$$p(f) = \sum_{k=-\infty}^{+\infty} \gamma_k \cdot \cos(2\pi f k), \; -1/2 \leqslant f \leqslant 1/2, \tag{3.4.19}$$

then we see that

$$\gamma_Z = p(0); \tag{3.4.20}$$

and the generalized analogue of the sample-size formula (3.4.8) is

$$N_2 = h^2(P^*, K) \cdot p(0)/(\Delta^*)^2 . \tag{3.4.21}$$

Thus to choose among K covariance stationary processes $(Z_{jt} : t > 0)$, $1 \leqslant j \leqslant K$, with indifference zone width Δ^* and probability of correct selection P^*, the following multiple ranking procedure is proposed:

<u>Procedure P_s</u>

1. Accumulate an initial series $(Z_{jt} : t = 1, \ldots, n_o)$ of length $n_o \geqslant 30$.
2. Compute the Heidelberger-Welch (1981) estimator $\hat{p}_j(0)$ of the *spectrum at zero frequency* using the series $(Z_{jt} : t = 1, \ldots, n_o)$.
3. Compute

$$n_j = \max \{ n_o, [h^2(n_o,P^*,K) \cdot \hat{p}_j(0)/(\Delta^*)^2] \}, \tag{3.4.22}$$

where $[y]$ denotes the smallest integer $\geqslant y$ and $h = h(n_o,P^*,K)$ is the unique solution of the equation

$$\int_{-\infty}^{+\infty} (F(z+h;\, n_o-1))^{K-1} \, f(z;n_o-1)dz = P^* \tag{3.4.23}$$

in which $f(\cdot;\nu)$ and $F(\cdot;\nu)$ respectively denote the PDF and CDF of a Student's - t variate with ν degrees of freedom.

4. If $n = n_o$, go to step 5. Otherwise, generate the next $n-n_o$ observations $(Z_{jt} : t = n_o + 1, \ldots, n)$ from the process.
5. Compute the usual sample mean from the entire series

$$Z_j = n_j^{-1} \sum_{t=1}^{n} Z_{tj} \tag{3.4.24}$$

and finally select the process yielding the maximal value $\bar{Z}_{[K]}$.

3.4.2 Calculation of Dudewicz-Dalal Critical Values

Development. It is awkward to require the user of procedure P_S to input the critical value $h(n_o, P^*, K)$ corresponding to the preselected levels of n_o, P^*, and K. Moreover, it is infeasible to enter the large table of critical values produced by Dudewicz, Ramberg, and Chen (1975) directly into the support procedure. Therefore, it was necessary to include within the MRP program a subroutine (RNKSEL) to solve equation (3.4.23) numerically.

In terms of the function

$$g(h) = \int_{-\infty}^{+\infty} (F(z+h;n_o-1))^{K-1} f(z;n_o-1)dz, \tag{3.4.25}$$

subroutine RNKSEL is designed to find the root of the equation $g(h) = P^*$. To do this, RNKSEL first must determine limits of integration, a and b, such that the function

$$g^*(h) = \int_{a}^{b} (F(z+h;n_o-1))^{K-1} f(z;n_o-1)dz, \tag{3.4.26}$$

satisfies

$$|g(h) - g^*(h)| < 10^{-6} \quad \text{for all h.} \tag{3.4.27}$$

Using the IMSL routine MDSTI with a value of 10^{-6} as the total area allowed in both tails of the Student's - t distribution with $n_o - 1$ degrees of freedom, RNKSEL obtains a cutoff value b with upper tail area equal to 0.5×10^{-6}. The lower limit a is set equal to -b. This ensures that equation (3.4.27) is satisfied. The IMSL routine ZSCNT is then used to find the root of

$$g^*(h) - P^* = 0 \tag{3.4.28}$$

Routine ZSCNT is based on the secant method for solving simultaneous equations (Wolfe, 1959). To do the required integration shown in equation (3.4.26), a cautious adaptive Romberg extrapolation technique of numerical integration is used (De Boor, 1971). For this purpose, the IMSL routine DCADRE is invoked with absolute and relative estimation errors both set to 10^{-8}. A listing of RNKSEL is included as a subroutine in Appendix C, and a flowchart is depicted in Figure 3.10.

Verification and Validation. This portion of the overall MRP analysis procedure encompasses the routines to calculate the necessary discrete-event simulation run length and the final steady-state performance statistic. This performance statistic can then be compared to a similar measure of performance for the other alternative systems in order to select the "best" alternative.

The program DND that was designed to perform these calculations has the following inputs:

1. P^*, Δ^*, n_o, and K specified by the user

2. A simulation generated time series that has been:

   a. Batched to induce approximate normality using program WILK

   b. Truncated to eliminate any initialization bias using program IBZERO

   c. Analyzed by the Heidelberger-Welch procedure to estimate the spectrum of the batched process at zero frequency (using program WELCH; see section 3.4.3).

The foundation on which this program rests is the code written to implement the Dudewicz and Dalal (1975) procedure P_E as described in section 2.1.2. Therefore, the starting point was to code this procedure and check it against the numerical examples provided by Dudewicz, Ramberg, and Chen (1975). This was completed, and in each case DND produced results that are identical to those reported in the literature.

The next logical test was to generate random samples from a set of normal populations with known parameters in order to verify the performance of the ranking procedure. The sample data sets were generated by the IMSL routine GGNML from populations arranged in a variety of least favorable configurations. Table 3.3 displays the results of 3 separate experiments within each of which the LFC was fixed and the exact Dudewicz-Dalal procedure P_E was replicated 100 times. Note that Table 3.3 reveals no significant departures from the nominal probability of correct selection P^* specified for each experiment.

Table 3.3 Performance of exact Dudewicz-Dalal procedure program DND for independent normal samples.

TEST NUMBER	ALTERNATIVE NUMBER j	μ_j	σ_j	P^*	Δ^*	n_o	h	TIMES SELECTED "BEST"
1	1	55	4.5	0.9	5	7	3.312	2
1	2	60	4.0					92
1	3	55	5.5					0
1	4	55	2.8					3
1	5	55	6.8					0
1	6	55	10.3	0.9	5	7	3.312	3
2	1	53	4.5	0.8	7	7	2.566	1
2	2	60	4.0					83
2	3	53	5.5					5
2	4	53	2.8					5
2	5	53	6.8					1
2	6	53	10.3	0.8	7	7	2.566	6
3	1	53	4.5	0.95	7	30	3.463	0
3	2	53	9.5					0
3	3	53	3.4					2
3	4	53	10.3					1
3	5	53	6.8					0
3	6	53	2.8					0
3	7	53	5.5					1
3	8	60	4.0	0.95	7	30	3.463	96

Up to this point, the critical value $h(n_o, P^*, K)$ was treated as a required input variable for the program DND. To eliminate this input requirement, the subroutine RNKSEL (described in section 3.4.2) was coded and integrated with DND. The resulting program, AUTOH, was exercised through a wide range of possible values of the input parameters n_o, P^*, and K; the resulting critical values were then verified item-by-item against the tables computed by Dudewicz, Ramberg, and Chen (1975).

Further testing of AUTOH against DND involved the use of common random numbers to re-create the experiments described in Table 3.3. Using the same random number seeds for corresponding runs of DND and AUTOH ensured that exactly the same sets of data were generated for the comparison. AUTOH successfully reproduced the results shown in Table 3.3. The detailed output produced by AUTOH for the second experiment is shown in Figure 3.9. A listing of AUTOH is shown in Appendix C. The AUTOH flowchart is Figure 3.10.

Up to this point, all of the testing had been limited to independent normal samples. The next logical step was the implementation of procedure P_s to handle correlated data. The main prerequisite for this step of the research was the development of a support routine to estimate the spectrum at zero frequencv.

3.4.3 Spectral Analysis Procedure

Development. To determine the spectral density at zero frequency, the first step is to calculate the fast Fourier transform

```
ALTERNATIVE   1 HAS   0 BEST RESULTS
THE TRUE MEAN IS   53.00 WITH A STANDARD DEVIATION OF    4.50

ALTERNATIVE   2 HAS  83 BEST RESULTS
THE TRUE MEAN IS   60.00 WITH A STANDARD DEVIATION OF    4.00

ALTERNATIVE   3 HAS   5 BEST RESULTS
THE TRUE MEAN IS   53.00 WITH A STANDARD DEVIATION OF    5.50

ALTERNATIVE   4 HAS   5 BEST RESULTS
THE TRUE MEAN IS   53.00 WITH A STANDARD DEVIATION OF    2.80

ALTERNATIVE   5 HAS   1 BEST RESULTS
THE TRUE MEAN IS   53.00 WITH A STANDARD DEVIATION OF    6.80

ALTERNATIVE   6 HAS   6 BEST RESULTS
THE TRUE MEAN IS   53.00 WITH A STANDARD DEVIATION OF   10.30

FOR THIS TEST: PCS =.800    DSEED =    .98413370D+07    DELTA = 7.00
   NAUGHT =    7   DNDH =2.566   NMAX =    24
```

Figure 3.9 Detailed output of program AUTOH for one experiment.

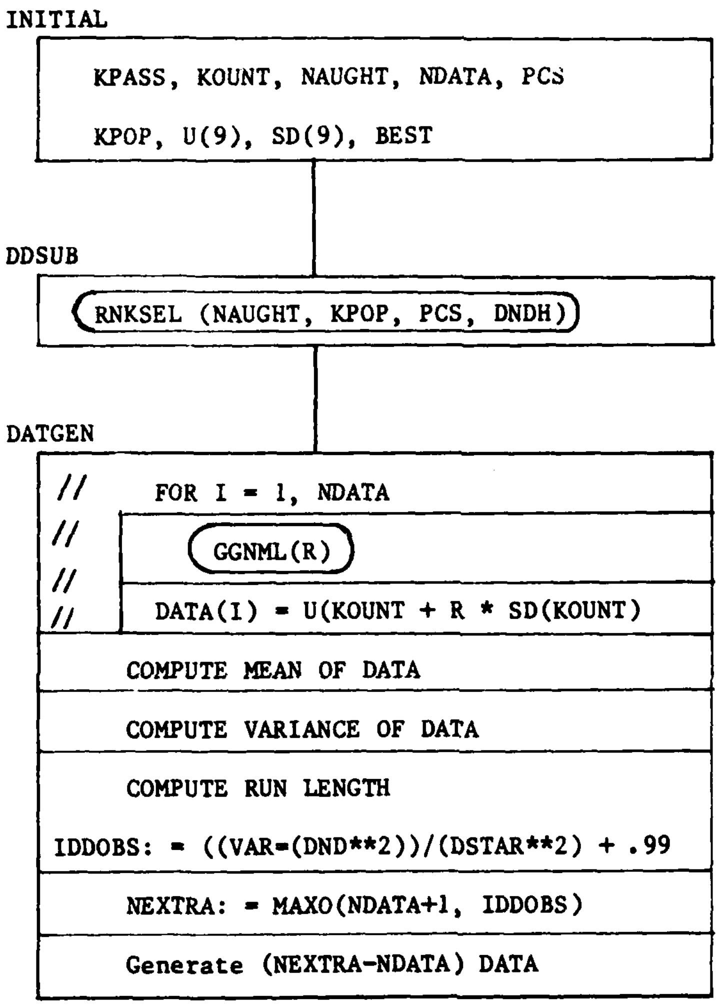


Figure 3.10 Detailed output of subroutine AUTOH for one experiment.

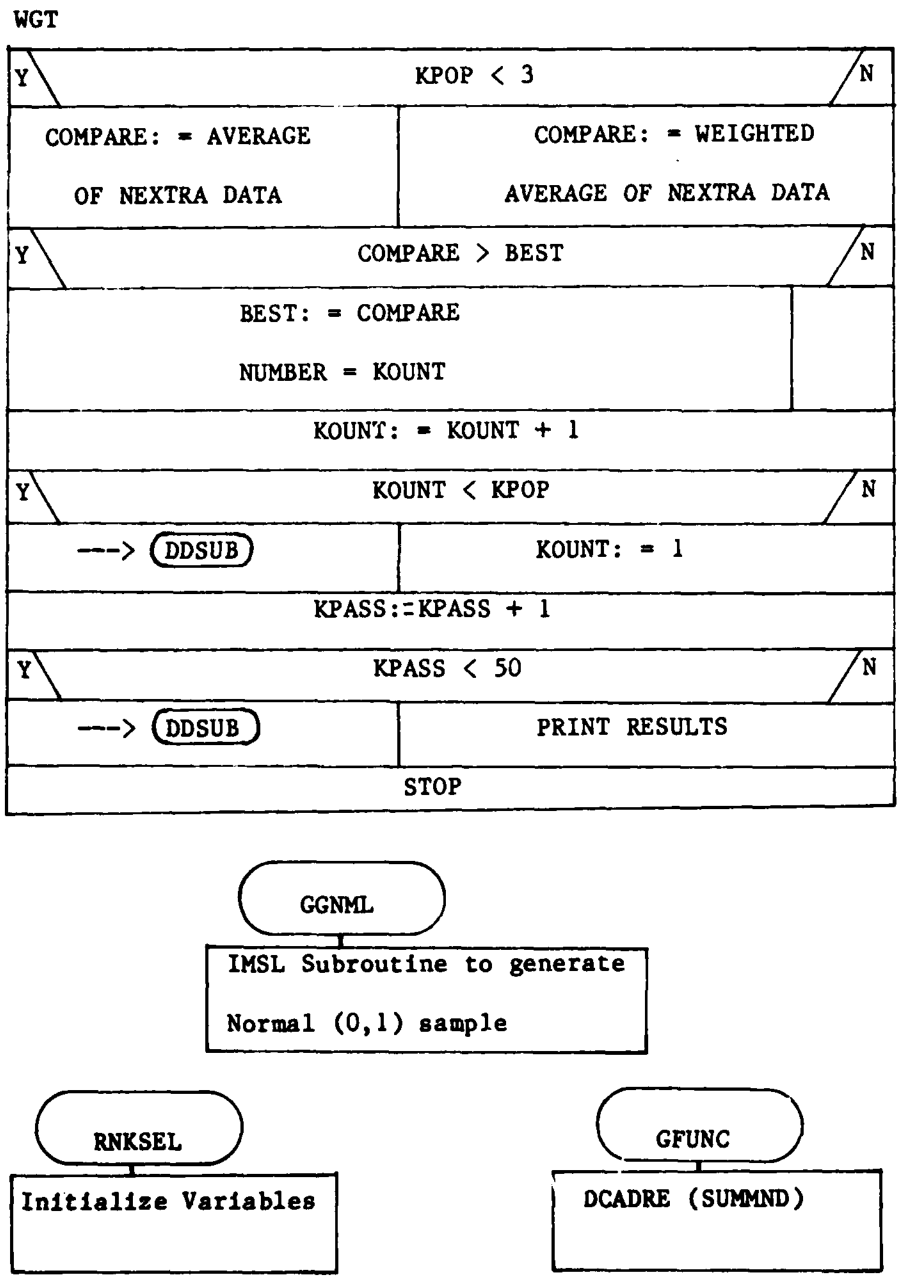


Figure 3.10 (continued)

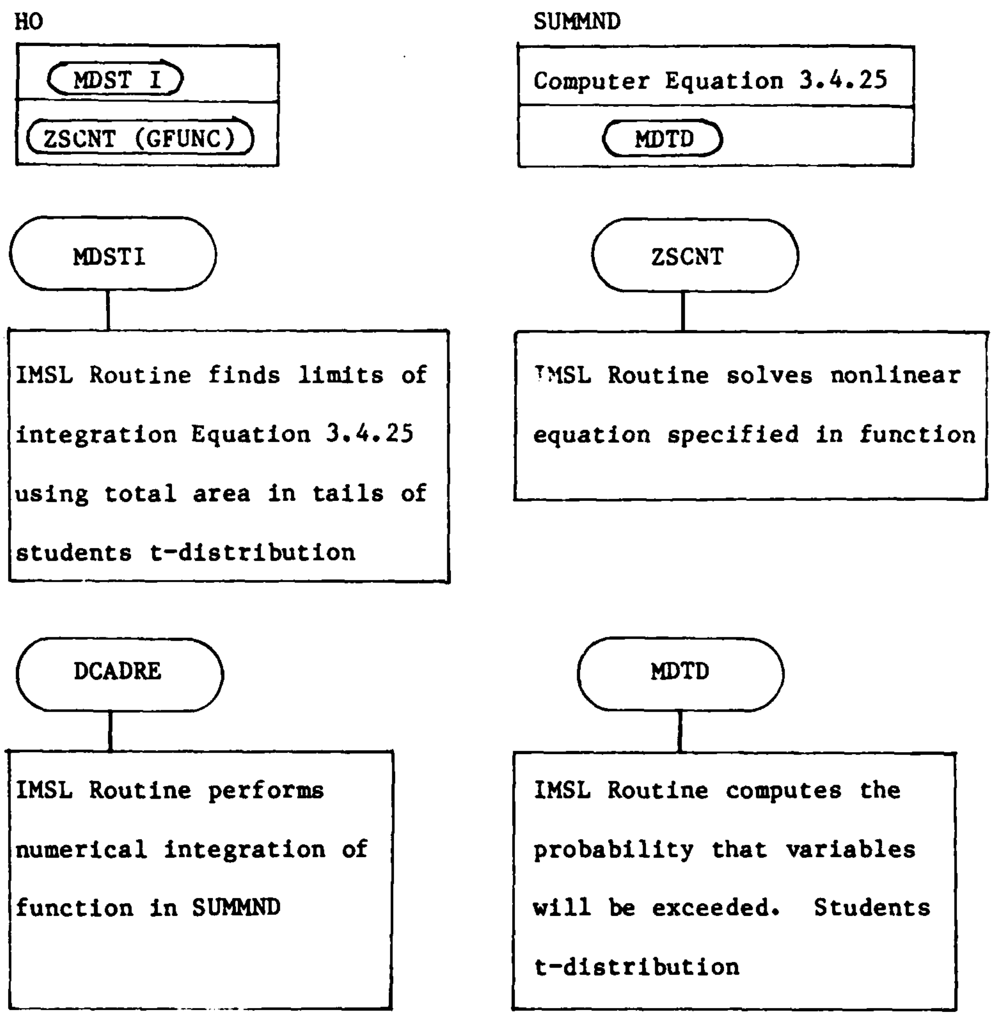
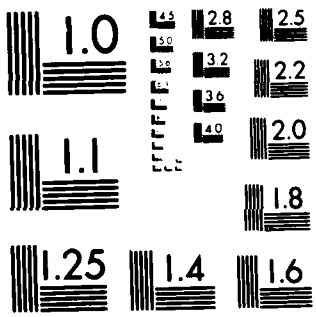


Figure 3.10 (continued)

AD-A133 649   A MULTIPLE RANKING PROCEDURE ADAPTED TO DISCRETE-EVENT SIMULATION(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH R T DICKINSON DEC 83 AFIT/CI/NR-83-47D   2/3

UNCLASSIFIED   F/G 12/1   NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963 A

of the observed series (Z_t : t = 1, ..., n)

$$d_u = \sum_{t=1}^{n} Z_t \cdot \exp[-i2\pi(t-1)(u-1)/n], \quad u = 1, \ldots, n, \qquad (3.4.29)$$

where $i = (-1)^{1/2}$. The periodogram defined by equation (2.4.7) is then given by

$$I(s/n) = \left| d_{s+1} \right|^2/n, \quad s = 0, 1, \ldots, n-1. \qquad (3.4.30)$$

The spectral analysis procedure WELCH actually invokes the IMSL routine FFTRC to compute the complex conjugate ($\bar{d}_u$: u = 1, ..., n) of the fast Fourier transform. Since the absolute value of a complex number is invariant under conjugation, this complication does not affect the computation (3.4.30) of the periodogram.

To stabilize the variance so that a polynomial can be fitted by the method of ordinary least squares, the periodogram is averaged over adjacent values and a logarithmic transformation is applied; then the constant 0.270 is added to eliminate the bias induced by the logarithmic transformation:

$$Y_u = 0.27 + \ln([I((2u-1)/n) + I(2u/n)]/2),$$
$$u = 1, \ldots, n/4 \quad . \qquad (3.4.31)$$

Corresponding to the u^{th} observation Y_u of the dependent variable, there can be up to d = 5 independent variables of the form

$$X_{uk} = [(4u-1)/(2n)]^k, \quad k = 1, \ldots, d. \qquad (3.4.32)$$

Program WELCH uses a forward stepwise algorithm, IMSL routine RLSEP, to find the best regression model of the form

$$Y_u = \sum_{k=0}^{d} a_k X_{uk} + \varepsilon_u, \qquad u = 1, \ldots, n/4. \qquad (3.4.33)$$

Parameters are set in RLSEP so that the significance level for entering and leaving variables is 0.05, and a partial F-test is performed for each term in the model. From the final regression model (3.4.33) selected by RLSEP, the estimated intercept $\hat{a}_0$ must be taken together with the final design matrix $\underset{\sim}{X} = \|X_{uk}\|$ to estimate the spectral density at zero frequency:

$$\hat{p}(0) = \exp(\hat{a}_0 - 0.3225[(\underset{\sim}{X}'\underset{\sim}{X})^{-1}]_{11}) \qquad (3.4.34)$$

Unfortunately RLSEP does not produce the upper left most element of $(\underset{\sim}{X}'\underset{\sim}{X})^{-1}$ as an ancillary output. To obtain this element, the final independent variables chosen by RLSEP are used as inputs to another IMSL regression analysis routine, RLMUL, after the required conditioning has been performed by routine BECOVM. RLMUL performs the computations required for a standard multiple linear regression analysis and supplies as outputs the values for the residual mean square, MS_E, and the estimated standard error of the intercept, $SE(\hat{a}_0)$. The required element is given by

$$(\underset{\sim}{X}'\underset{\sim}{X})^{-1} = [SE(\hat{a}_0)]^2/MS_E \ . \qquad (3.4.35)$$

The overall flowchart of program WELCH is shown in Figure 3.11. The program listing is presented in Appendix D.

Verification and Validation. During the development of program WELCH, several intermediate tests were performed. Specifically, the integration of the IMSL regression routines RLSEP and RLMUL was tested on the well-known Hald data (Draper and Smith,

```
Initialize variables U, SD
DO J: = 1,25
    DO K: = 1, NDATA
        GGNML(R)
        DATA(K): = R(J) * SD + U
    NHDATA: = NDATA/2
    NQDATA: = NQDATA/2
    FFTRC(DATA, TRANS)
    COMPUTE PERIODOGRAM
    PERIOD (L)
    SMOTH: = ((PERIOD(L) + PERIOD (L+1))/2)
    FJ(L): = ALOG(SMOTH) + .270
    RLSEP(FJ)
    BECOVM
    RLMUL(FJ)
    DETERMINE UPPER LEFT ELEMENT (UPLEFT)
    IN REGRESSION MATRIX
    CONE: = EXP(- 0.3225 * UPLEFT)
    COMPUTE SPECTRAL DENSITY AT ZERO FREQUENCY
    COMPUTE CLASSICAL VARIANCE ESTIMATE
PRINT RESULTS
```

Figure 3.11 Flowchart of WELCH for estimation of the spectrum at zero frequency.

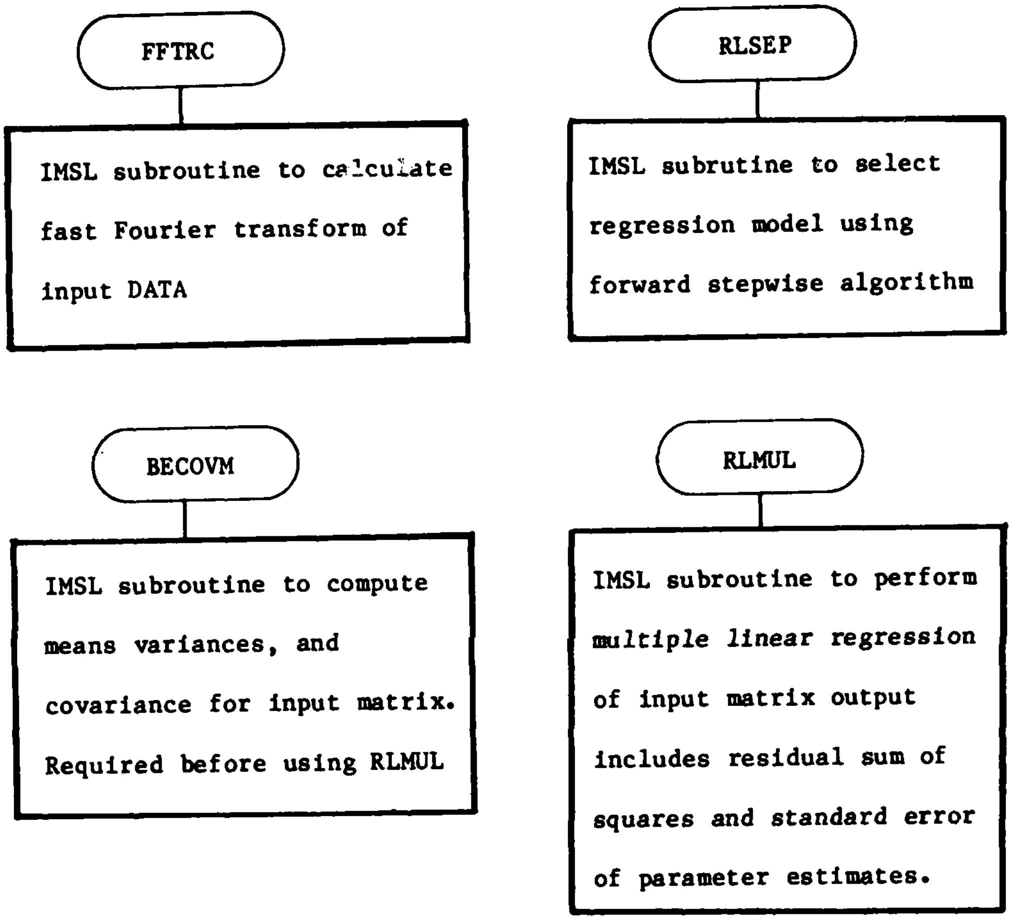


Figure 3.11 (continued)

1967, p. 164). The final model obtained by WELCH coincides with the results found by Draper and Smith (1967, Appendix B).

The first overall test of the spectral-estimation logic used independent identically distributed normal variates. In each of 5 different experiments, a normal population was specified and the IMSL routine GGNML was used to generate 25 random samples of size 100 from that population. For each sample, both the sample variance and the estimated spectrum at zero frequency were calculated. These estimates were then averaged over each experiment. The results are shown in Table 3.4.

Table 3.4 Performance of spectral-estimation routine WELCH for independent normal samples.

POPULATION PARAMETERS		SAMPLE ESTIMATES	
μ	σ^2	S^2	$\hat{p}(0)$
50	25	25.1	25.6
22	25	25.5	26.6
50	900	917.1	956.3
10	36	36.4	37.9
5	36	38.0	40.0

Following this comparison, program WELCH was tested on correlated data. All test series were generated by the previously

discussed program ARMAPQ using autoregressive and moving-average parameters that were selected to ensure stationarity and invertibility (Box and Jenkins, 1970, Charts B, C, and D). The theoritical value of the spectrum at zero frequencv was calculated using equation (3.4.17). Table 3.5 shows the parameters used to generate each ARMA process, the theoretical value p(0), the estimate $\hat{p}(0)$ produced by WELCH, and the sample size n.

3.4.4 Integrated Testing of Multiple Ranking Procedure

Up to this point we have demonstrated separately the successful implementation of the exact Dudewicz-Dalal procedure P_E (program DND), the automatic determination of the critical value $h(n_o$ P^*, K) (program AUTOH), and the estimation of the spectrum at zero frequency p(0) (program WELCH). To develop a program to perform the extended multiple ranking procedure P_S requires

1. The integration of AUTOH and WELCH
2. The elimination of the weighting scheme (2.1.16) within AUTOH.

The final integrated package was called NOWAIT.

As with the previously discussed programs, the first overall test of NOWAIT used independent normally distributed data sets representing K alternatives with the means arranged in a least favorable configuration. The parameters of the selected normal populations are shown in Table 3.6. The NOWAIT procedure was replicated 100 times using this configuration of normal populations. The overall results for this experiment are shown in Figure 3.12. The

Table 3.5 Performance of spectral-estimation routine WELCH for ARMA series.

	PROCESS PARAMETERS ϕ_1	ϕ_2	θ_1	θ_2	μ	σ_a^2	THEORETICAL VALUE p(0)	ESTIMATE $\hat{p}(0)$	SAMPLE SIZE
1	-	-	0.25	-	100	225	126.6	132	200
2	-	-	-0.4	-	100	225	441	540.8	200
3	-	-	-0.1	-	100	225	272.2	263.8	200
4	-	-	0.9	-	100	225	2.25	3.6	200
5	-	-	0.9	-	100	225	2.25	3.4	500
6	-0.2	-	-	-	10	9	6.25	5.9	200
7	0.6	-	-	-	10	9	56.2	50.1	200
8	0.1	-	-	-	10	9	11.1	9.2	200
9	-0.8	-	-	-	10	9	2.8	2.9	200
10	-0.5	-	0.15	-	100	400	128.4	125.7	200
11	0.6	-	0.2	-	100	144	576	704.8	200
12	0.75	-	-0.5	-	100	400	14400	17003	200
13	-0.6	-	0.5	-	100	144	14.1	10.7	200
14	0.6	-	0.2	-	100	400	1600	1710	200
15	-	-	0.25	-0.75	160	900	2025	3573	200
16	-	-	0.25	-0.75	160	900	2025	2651	500
17	-	-	-0.5	0.15	160	900	1640.2	1714.2	500
18	-	-	0.75	-0.5	160	900	506.2	866.7	500
19	0.65	0.1	-	-	100	400	6400	7933	200
20	0.65	0.1	-	-	100	400	6400	5730.5	500
21	-0.35	0.25	-	-	100	400	330.6	243.4	500
22	-0.25	-0.5	-	-	100	400	130.6	122.3	500
23	-	-	-	-	10	9	9	8.6	200

Table 3.6 Configuration of independent normal samples for testing the integrated package NOWAIT.

ALTERNATIVE	POPULATION PARAMETERS	
	μ	σ
1	53	8.5
2	60	8.0
3	53	10.1
4	53	5.6
5	53	12.8
6	53	20.3
7	53	3.4
8	53	9.5
9	53	8.6

```
ALTERNATIVE   1 HAS   0 BEST RESULTS
THE TRUE MEAN IS   53.00 WITH A STANDARD DEVIATION OF    8.50

ALTERNATIVE   2 HAS  96 BEST RESULTS
THE TRUE MEAN IS   60.00 WITH A STANDARD DEVIATION OF    8.00

ALTERNATIVE   3 HAS   0 BEST RESULTS
THE TRUE MEAN IS   53.00 WITH A STANDARD DEVIATION OF   10.10

ALTERNATIVE   4 HAS   0 BEST RESULTS
THE TRUE MEAN IS   53.00 WITH A STANDARD DEVIATION OF    5.60

ALTERNATIVE   5 HAS   1 BEST RESULTS
THE TRUE MEAN IS   53.00 WITH A STANDARD DEVIATION OF   12.80

ALTERNATIVE   6 HAS   3 BEST RESULTS
THE TRUE MEAN IS   53.00 WITH A STANDARD DEVIATION OF   20.30


FOR THIS TEST: PCS =.950    DSEED =    .17500000D+03    DELTA = 7.00
   NAUGHT =  30   DNDH =3.297    NMAX =   93
```

Figure 3.12 Output for independent normal test of the integrated package NOWAIT.

variable NMAX shown in Figure 3.12 is the largest sample size needed on any replication of the procedure. The critical value $h(n_o = 30, P^* = 0.95, K = 6)$ is shown to be approximatelv 3.297 for this particular test.

An additional test using independent normal samples was performed with 1 of 9 population means lying within the indifference zone. The results shown in Figure 3.13 reveal that while the program did not pick the "best" alternative on 90% of the replications, it did pick a population lying within the indifference zone on 99% of the replications.

The final check of program NOWAIT used covariance stationary series generated by the subroutine ARMAPQ as previously discussed. The mean values of these ARMA processes were established to produce a series of alternatives arranged in a least favorable configuration. The test series included AR(1), AR(2), MA(1), MA(2), and ARMA(1,1) models. The result of this testing is shown in Table 3.7. Included in the table are the parameters used for the ranking procedure and the maximum sample size required. The percentage of correct selections is based on fifty independent replications of a particular configuration of ARMA processes. Table 3.8 shows the autoregressive and moving average parameters used to generate each of the test series. These results provide good evidence of the utility of the multiple ranking procedure P_s for covariance stationary processes.

```
ALTERNATIVE   1 HAS   0 BEST RESULTS
THE TRUE MEAN IS   53.00 WITH A STANDARD DEVIATION OF    8.50

ALTERNATIVE   2 HAS   0 BEST RESULTS
THE TRUE MEAN IS   53.00 WITH A STANDARD DEVIATION OF    8.00

ALTERNATIVE   3 HAS   0 BEST RESULTS
THE TRUE MEAN IS   53.00 WITH A STANDARD DEVIATION OF   10.10

ALTERNATIVE   4 HAS   0 BEST RESULTS
THE TRUE MEAN IS   53.00 WITH A STANDARD DEVIATION OF    5.60

ALTERNATIVE   5 HAS   1 BEST RESULTS
THE TRUE MEAN IS   53.00 WITH A STANDARD DEVIATION OF   12.80

ALTERNATIVE   6 HAS   0 BEST RESULTS
THE TRUE MEAN IS   53.00 WITH A STANDARD DEVIATION OF   20.30

ALTERNATIVE   7 HAS   0 BEST RESULTS
THE TRUE MEAN IS   53.00 WITH A STANDARD DEVIATION OF    3.40

ALTERNATIVE   8 HAS  16 BEST RESULTS
THE TRUE MEAN IS   58.00 WITH A STANDARD DEVIATION OF    9.50

ALTERNATIVE   9 HAS  83 BEST RESULTS
THE TRUE MEAN IS   60.00 WITH A STANDARD DEVIATION OF    8.60


FOR THIS TEST: PCS =.900    DSEED =    .56732100D+06   DELTA = 7.00
   NAUGHT =  30   DNDH =3.051   NMAX =  159
```

Figure 3.13 Output for independent normal test of NOWAIT with one mean in the indifference zone.

Table 3.7 Performance of the integrated package NOWAIT for ARMA series.

Initial Sample Size N_o	Indifference Zone Width Δ^*	Alternatives Number K	Best	Maximum Required Sample N_{max}	Probability Of Correct Selection Pr(CS)	% Of Correct Selections
30	20	3	3	106	0.95	0.98
30	10	3	3	422	0.95	0.94
50	20	3	3	242	0.95	1.00
50	10	3	3	542	0.95	0.94
30	20	5	2	193	0.85	1.00
30	20	5	4	108	0.85	0.98
30	15	5	4	192	0.85	0.92
30	10	5	4	430	0.85	0.82
100	10	5	4	277	0.85	0.96
100	10	5	4	494	0.95	0.98
20	20	7	5	180	0.95	0.92
20	20	7	6	180	0.95	0.94
30	10	7	4	519	0.85	0.78
30	20	7	6	100	0.95	1.00
30	20	7	5	423	0.95	0.98
50	10	7	4	667	0.95	0.90
50	10	7	4	647	0.85	0.88
100	10	7	4	653	0.95	0.98

Table 3.8 Configuration of ARMA processes for testing the integrated package NOWAIT.

ALTERNATIVE	MODEL	ϕ_1	ϕ_2	θ_1	θ_2	σ_a^2	p(0)
1	AR(1)	0.6				400	2500
2	AR(2)	-0.35	0.25			625	516
3	MA(1)			-0.4		400	784
4	MA(2)			0.75	-0.5	1225	689
5	ARMA(1,1)	-0.5		0.15		1600	514
6	ARMA(1,1)	0.6		0.2		169	676
7	ARMA(1,1)	0.75		-0.5		49	1764

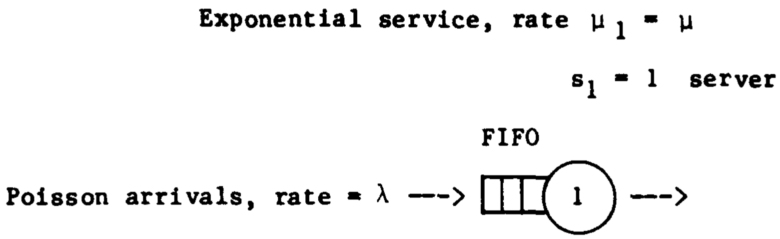
CHAPTER IV

EXPERIMENTAL RESULTS

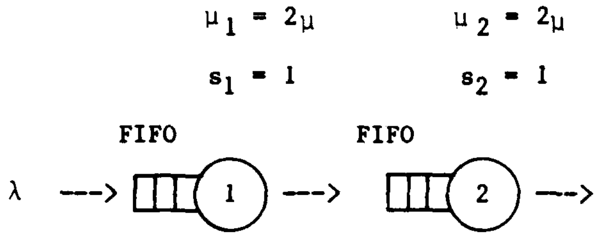
Although the autoregressive-moving average processes described in Chapter III provide an appropriate means for verification and validation of procedure NOWAIT, such processes cannot adequately represent the full variety of transient and stationary time-series behavior characteristic of discrete-event simulations. This chapter presents the results of two meta-experiments that were specifically designed to evaluate the robustness of the multiple ranking procedure when it is applied to diverse simulation models. In the first meta-experiment, the procedure was applied to the customer sojourn time process in several tandem queueing systems with both high and low traffic intensities. The second meta-experiment focused on the series of yearly costs incurred during the operation of several (s,S) inventory systems.

4.1 Comparison of Tandem Queueing Systems

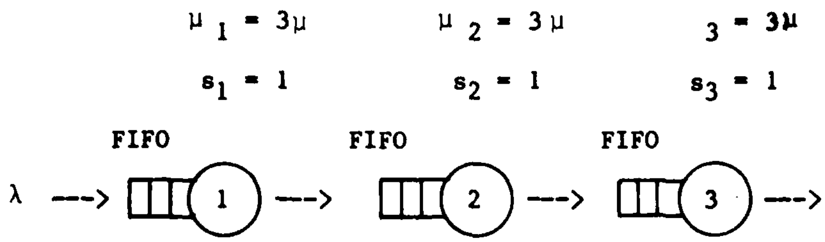
The first meta-experiment involves the comparison of three alternative configurations for a proposed repair facility. The layout for each alternative is shown in Figure 4.1. The calling units to be repaired arrive according to a Poisson process with rate λ. In system A_i ($i = 1, 2, 3$), there are i M/M/1 workstations in tandem; and service times at each workstation are IID exponential with mean $\mu_i^{-1} = (i\mu)^{-1}$. The traffic intensity $\rho = \lambda \cdot (i\mu_i^{-1}) = \lambda/\mu < 1$ ensures



Alternative A_1: M/M/1 queue.



Alternative A_2: Two M/M/1 queues in tandem.



Alternative A_3: Three M/M/1 queues in tandem.

Figure 4.1 Layout of tandem queueing systems compared in the first meta-experiment.

that system A_i is stable and thus has a limiting (steady-state) distribution for the time-in-system process (that is, for the sojourn times observed bv successive customers). It is desired to select the configuration whose mean sojourn time W_i is smallest. The alternative systems are to be tested at the traffic intensities $\rho = 0.5$ and $\rho = 0.8$.

Event-oriented models of these systems were implemented in the SLAM simulation language (Pritsker and Pedgen, 1979). Complete program listings are given in Appendix E. Although it is substantially easier to build a process interaction model of each alternative. it should be noted that the comparable event-oriented model requires significantly less execution time. Because of the scale of the experimentation performed in this research, execution efficiency was a critical factor in the choice of modeling technique.

Since there is no restriction on queue capacity at the repair stations, each station can be analyzed separatelv as a single-stage (nonseries) queueing model (Gross and Harris, 1974, p. 198). Additionally, with the first station being a M/M/1 queue, the steady state departure process has the same distribution as the interarrival time process. Therefore, all repair stations can be treated as M/M/1 queues; and the average time in system for the multistage queue is the appropriate multiple of the mean sojourn time in one M/M/1 queue:

$$W_i = i \cdot (\mu_i - \lambda)^{-1} = 1/(\mu - \lambda/i). \qquad (4.1.1)$$

If we take $\mu = 1$ so that $\lambda = 1/\rho$, then the mean sojourn times (W_i: $i = 1, 2, 3$) for the various alternatives are given in Table 4.1.

Table 4.1 Mean Sojourn times W_i for tandem queueing systems.

System	Traffic	Intensity
A_i	0.5	0.8
A_1	2.00	5.00
A_2	1.33	1.67
A_3	1.20	1.36

This analysis shows that configuration A_3, three fast servers, has the smallest average time in system and is therefore judged "best". The results in Table 4.1 were also used to determine the indifference zone for the multiple ranking procedure. The value of Δ^* was set so that the difference between the smallest mean sojourn time and the next smallest is greater than Δ^*:

$$W_{[2]} - W_{[1]} > \Delta^* . \tag{4.1.2}$$

Thus when $\rho = 0.5$, we took $\Delta^* = 0.1$; and when $\rho = 0.8$, we took $\Delta^* = 0.3$. For completeness, the first meta-experiment also included two levels for the probability of correct selection: $P^* = 0.90$ and $P^* = 0.95$.

Figure 4.2 is a flowchart of the protocol that was followed in the first meta-experiment. The steps of this protocol are enumerated below:

1. The alternative to be simulated is initialized in the "empty and idle" state. As each customer departs the last station,

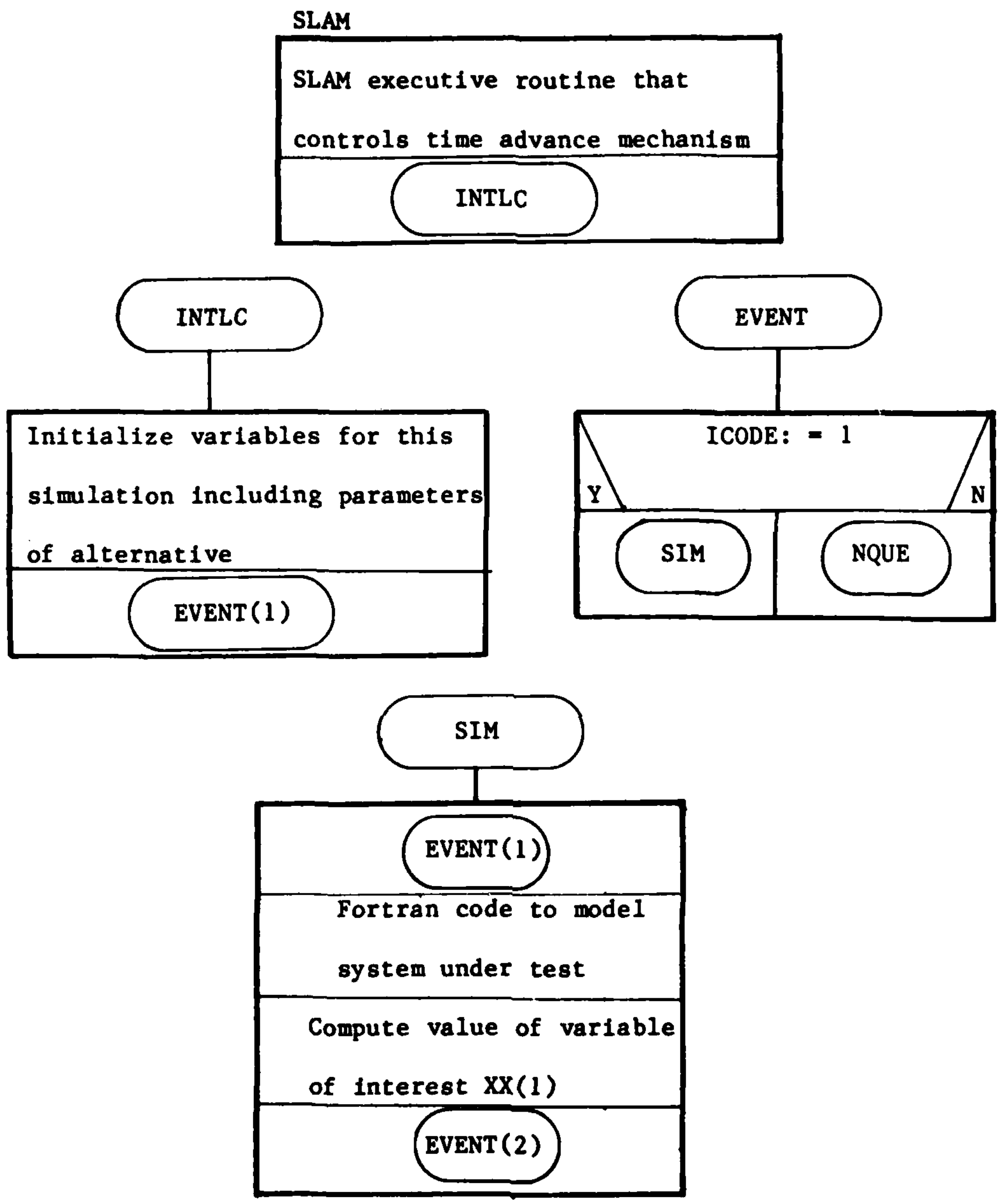


Figure 4.2 Flowchart of protocol for the first meta-experiment.

NQUE

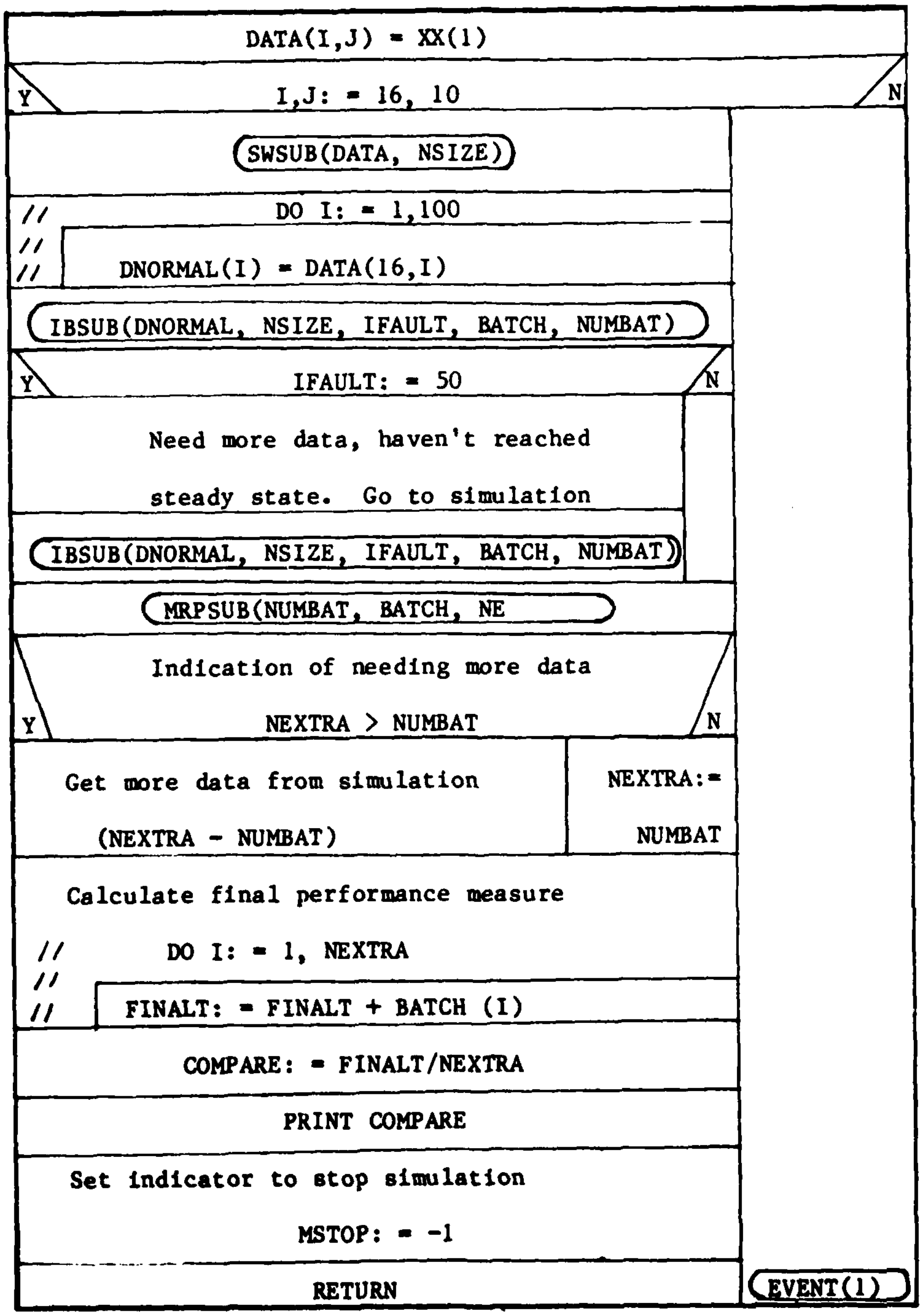


Figure 4.2 (continued)

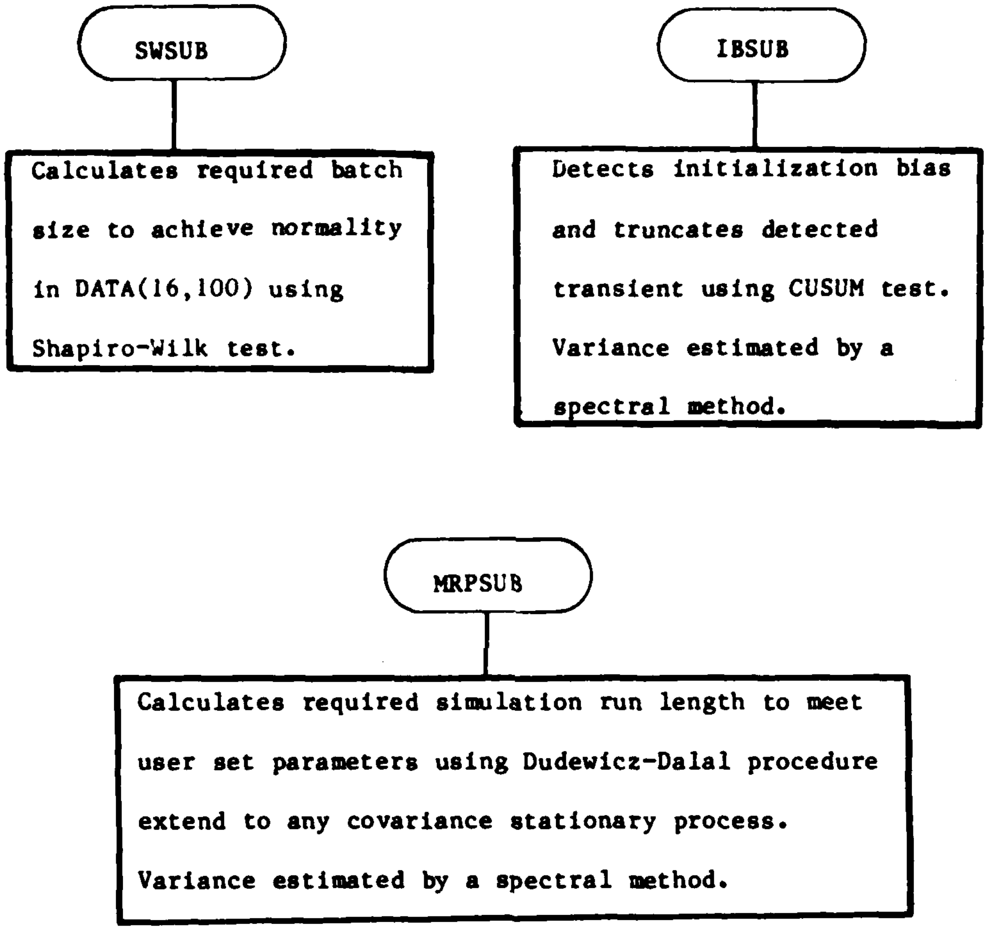


Figure 4.2 (continued)

his system residence time (waiting time plus service time) is recorded. The run is terminated after 100 sojourn times have been recorded.

2. The random number streams are independently reseeded and another run is executed. The simulation stops upon completion of 16 independent replications of 100 customers each.

3. Subroutine NQUE is called. To determine an adequate batch size (NSIZE), subroutine SWSUB is called by NQUE.

4. The data recorded during the last replication is grouped into $b = 100/\text{NSIZE}$ batches and is then tested for the presence of initialization bias by subroutine IBSUB. Appropriate data truncation is performed and the remaining set of n_o batch means is ready for further processing.

5. Subroutine MRPSUB is invoked. MRPSUB first calls subroutine RNKSEL to compute the Dudewicz-Dalal critical value $h(n_o, P^*, K)$. MRPSUB then calls subroutine WELCH to obtain the estimated spectrum at zero frequency $p(0)$. Finally, the required simulation run length n is computed from equation 3.4.22 .

6. The 16th replicate, which was suspended to perform steps 3 through 5 above, is now resumed. This run is terminated when $(n - n_o)$*NSIZE additional customers have been simulated.

7. The final estimator of W_i is the unweighted sample mean

$$\bar{W}_i = n^{-1} \sum_{j=1}^{n} W_{ij} \qquad (4.1.3)$$

of all the observations generated on the last replication of alternative A_1 (excluding the truncated observations).

8. Steps 1 through 7 are repeated 50 times, and each final performance statistic is stored. Each of the 50 experiments is started with a randomly selected random number seed. This ensures that the 50 experiments performed on each alternative are independent.

9. For each alternative under consideration, steps 1 through 8 are repeated.

10. The final results for each alternative are compared experimentwise — that is, for each experiment, the alternatives A_1, A_2, and A_3 are compared and the "best" alternative is selected. The final selections are tallied over all 50 experiments.

Table 4.2 summarizes the results of the first meta-experiment.

Table 4.2 Final results of the first meta-experiment.

Test Number	Random Number Stream Used	ρ	Δ^*	P^*	# Times Selected			% Correct Selections
					A_1	A_2	A_3	
1	4	0.5	0.1	0.95	0	1	49	98%
2	6	0.5	0.1	0.95	0	1	49	98%
3	3	0.8	0.3	0.90	0	1	49	98%
4	6	0.8	0.3	0.90	0	5	45	90%

The test results demonstrate that the integrated MRP does determine adequate simulation run-lengths. The resultant steady-state performance estimators of the form (4.1.3) appear to satisfy the nominal probability requirement; in every case the actual percentage of correct selections does not differ significantly from the nominal percentage P^*.

4.2 Comparison of (s,S) Inventory Systems

To examine the performance of the multiple ranking procedure under conditions radically different from those observed in queueing systems, we applied the procedure to $K = 6$ (s,S) inventory systems with probability requirement $P^* = 0.85$. For this situation, "best" refers to the system with the lowest expected annual operating cost. The following costs are associated with each inventory system:

$$\text{Ordering cost } OC = \$0.50/\text{order} \qquad (4.2.1)$$

$$\text{Holding cost } HC = \$0.10/\text{unit-week} \qquad (4.2.2)$$

$$\text{Shortage (backorder) cost } SC = \$1.00/\text{unit}. \qquad (4.2.3)$$

The (s,S) ordering policy operates as follows: if the number of units on hand at the end of the week is less than the reorder point s, an order is placed to bring the inventory position up to the stock control level S. (The inventory position is equal to the amount on hand plus the amount on order minus the amount backordered.) The order is filled immediately; thus when the store opens on Monday, the order placed on the preceding Friday has already arrived. Backorders

are permitted. The weekly demand is uniformly distributed on the range from 0 to 6 units. The fixed order charge OC is incurred no matter how many units are actually ordered. The shortage cost SC is charged for each unit backordered in the week that the backorder is placed. Each unit that must be carried over to the next week incurs the cost HC on Friday. Table 4.3 summarizes the six alternative (s,S) policies to be compared.

Table 4.3 Alternative (s,S) inventory systems compared in the second meta-experiment.

Alternative A_i	Policy (s_i,S_i)
A_1	(2,6)
A_2	(3,5)
A_3	(3,6)
A_4	(4,6)
A_5	(5,6)
A_6	(6,6)

A generalized, event-oriented simulation model for (s,S) inventory systems was coded in the SLAM simulation language. The portion of the program listing which differs from that of the queueing model is shown in Appendix F. The same protocol was used for the meta-experiment involving the inventory systems as was used for the tandem queueing systems.

The assumption which makes this problem analytically tractable

is that the starting-stock process (the inventory on hand at the beginning of the week <u>after</u> the previous order has been received) exhibits the Markovian property. This is equivalent to stating that the conditional probability of any future starting stock given a demand history and the present starting stock, is independent of the past demands and depends upon only the present starting stock. Thus the starting-stock process $(X_t: \ t = 0, 1, \ldots)$ is a finite-state Markov chain. If the weekly demand has probability density g_q, $q = 0, 1, \ldots$, then the chain (X_t) has the following transition probability matrix:

$$\underset{\sim}{P} = \begin{array}{c} \\ s \\ s+1 \\ \vdots \\ S-1 \\ S \end{array} \begin{array}{c} \begin{array}{ccccc} s & s+1 & & S-1 & S \end{array} \\ \begin{bmatrix} g_0 & 0 & \cdots & 0 & \sum_{q>0} g_q \\ g_1 & g_0 & \cdots & 0 & \sum_{q>1} g_q \\ & & & & \\ g_{S-s-1} & g_{S-s-2} & \cdots & g_0 & \sum_{q>S-s-1} g_q \\ g_{S-s} & g_{S-s-1} & \cdots & g_1 & g_0 + \sum_{q>S-s} g_q \end{bmatrix} \end{array} \qquad (4.2.4)$$

In particular, the (4,6) policy (alternative A_4) has the following one-step transition matrix:

$$\underset{\sim}{P}(A_4) = \begin{array}{c} \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{ccc} 4 & 5 & 6 \end{array} \\ \begin{bmatrix} .1429 & 0 & .8571 \\ .1429 & .1429 & .7142 \\ .1429 & .1429 & .7142 \end{bmatrix} \end{array} \qquad (4.2.5)$$

The vector of steadv-state probabilities

$$\underset{\sim}{\pi} = [\,\pi_s,\ \pi_{s+1},\ \cdots,\ \pi_S\,] \tag{4.2.6}$$

is found by solving the following system of equations:

$$\left.\begin{aligned} \underset{\sim}{\pi} &= \underset{\sim}{\pi}\,\underset{\sim}{P} \\ \underset{\sim}{\pi} \cdot 1 &= 1 \end{aligned}\right\} . \tag{4.2.7}$$

For example, in the (4,6) inventory system (alternative A_4), the steady-state distribution of the starting stock is: $\pi_4 = 0.1429$, $\pi_5 = 0.1224$, $\pi_6 = 0.7347$.

The expected value of the weekly cost process (C_t: t = 1, 2, ...) can be computed by a straightforward application of the law of total probability:

$$\begin{aligned} E(C_t) &= \sum_{j=s}^{S} \pi_j \cdot \Sigma[C_t \mid X_t = j] \\ &= OC \cdot \sum_{j=s}^{S} \pi_j \cdot \left(\sum_{q=j-s+1}^{\infty} g_q \right) \\ &+ HC \cdot \sum_{j=s}^{S} \pi_j \cdot \left[\sum_{q=0}^{j} (j-q) g_q \right] \\ &+ SC \cdot \sum_{j=s}^{S} \pi_j \cdot \left[\sum_{q=j}^{\infty} (q-j) g_q \right] . \end{aligned} \tag{4.2.8}$$

Table 4.4 summarizes the expected yearly operating costs for each of the alternative inventory policies.

Table 4.4 Expected yearly operating costs for (s,S) inventory systems.

Inventory Policy	Expected Yearly Costs
(2,6)	$ 42.17
(3,5)	40.66
(3,6)	36.95
(4,6)	34.35
(5,6)	35.13
(6,6)	37.89

As in the case of the repair facility simulation, each alternative inventorv policy was subjected to 50 independent multiple-ranking experiments. Each experiment consisted of 16 independent replications of 100 years of simulated operation; the final replication was then continued to yield the sample size required bv the multiple ranking procedure. The overall results of the second meta-experiment are summarized in Table 4.5.

Table 4.5 Final results of the second meta-experiment.

Random Number Stream Used	n_o	Δ^*	P^*	Number of Times Selected (2,6)	(3,5)	(3,6)	(4,6)	(5,6)	(6,6)
7	100	0.75	0.85	0	0	0	50	0	0
7	30	0.75	0.85	0	0	0	48	2	0

The cost process (C_t) in an (s,S) inventory system exhibits a correlation structure which is fundamentally different from that of the sojourn time process in an M/M/1 queue. This is shown graphicallv in Figures 4.3 and 4.4. Thus whereas the sample variance of n successive M/M/1 sojourn times seriously underestimates the variance parameter

$$\gamma_W = n \cdot \lim_{n \to \infty} \mathrm{Var}(\overline{W}_n) = p_W(0)$$

the sample variance of n successive weekly inventory costs overestimates the quantity

$$\gamma_C = n \cdot \lim_{n \to \infty} \mathrm{Var}(\overline{C}_n) = p_C(0) \quad .$$

The implication is that reduced sample sizes can be achieved with a multiple ranking procedure for inventorv cost processes based on a reliable estimator $\hat{p}_C(0)$ of the corresponding spectrum at zero frequency. To show this, the same set of inventory simulation experiments was rerun with the initial sample size n_o reduced from 100 to 30. The results are included in Table 4.5.

In all simulations of the (s,S) inventory system, the model was initialized with S units in the inventory. Initialization bias was detected in onlv one of the 100 experiments that were performed. This is another manifestation of the diversity stochastic behavior achieved by the use of both queueing and inventory models.

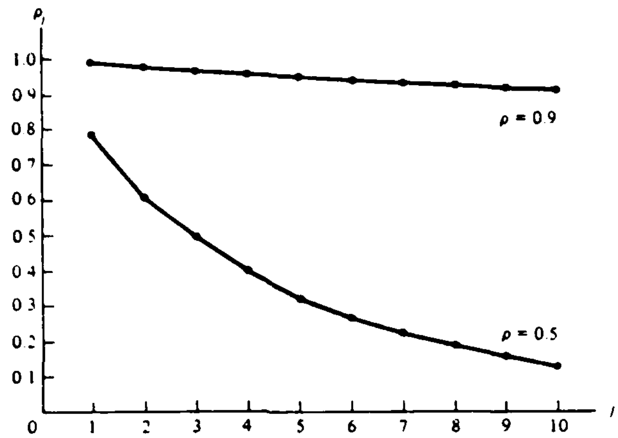


Figure 4.3 Correlation function of the sojourn time process for an M/M/1 queue.

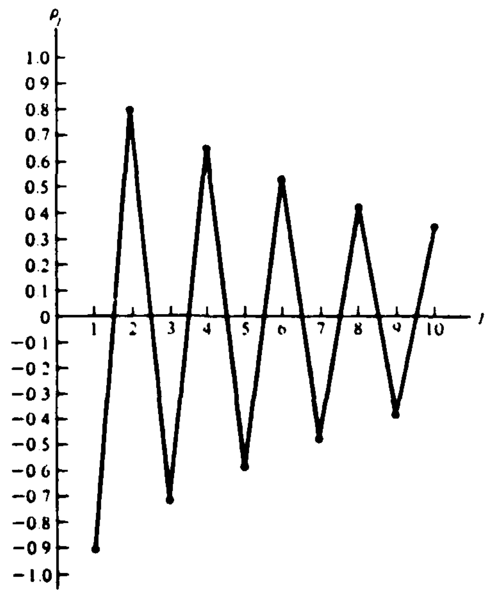


Figure 4.4 Correlation function of the weekly cost process for an (s,S) inventory system.

CHAPTER V

SUMMARY AND RECOMMENDATIONS

5.1 Main Findings of the Research

If the practitioner uses the procedures and programs developed in this research to control the run-length of a discrete-event simulation, he will be able to select the "best" of K competing alternative configurations efficiently and reliably. The user is able to set the probability of correct selection and the indifference zone width to meet his needs. The only assumption required is that the output of the model should approach a covariance stationary process as the run length increases.

The Shapiro-Wilk test has proved to be an effective means for determining a batch size sufficient to induce an acceptable degree of convergence to normality in simulation-generated output series. The cusum process defined on the resulting sequence of batch means can then be used both to detect the presence of initialization bias and to eliminate the bias economically. The initialization bias test procedure developed in this research has demonstrated its power against a wide range of alternatives, yet the procedure is not dependent on the particular form of the bias function. There is no empirical evidence that the cusum procedure causes either false indications of initial bias or excessive data truncation.

The main objective of this research was to develop a generalized multiple ranking procedure for covariance stationary

processes. The Dudewicz-Zaino procedure for AR(1) processes was shown to be a special case of the procedure developed in this research. Equally important to the potential user is the fact that this generalized procedure is based on a well-established estimator of the spectral density at zero frequency. This spectral method provides a simple and robust means of capturing the relevant information about the covariance structure of simulation-generated output processes.

The successful application of the integrated multiple ranking procedure to systems whose outputs exhibit widely differing types of stochastic behavior demonstrates the utility of the procedure in the analysis of discrete-event simulation models.

5.2 Recommendations for Future Research

The results of this research indicate the need for further work in a number of areas. The requirement that the output process be approximately Gaussian warrants additional investigation. Using a maximum batch size of 3 in the Shapiro-Wilk test does not seem to cause any serious problems in subsequent stages of the multiple ranking procedure. This casts some doubt on the necessity for normally distributed observations. Clearly, we require a more precise understanding of the role of batching in guaranteeing the reliability of the multiple ranking procedure.

While the initialization bias test procedure proved to be effective for a broad range of transient processes, one deficiency is still evident. The difficulty of identifying a gradually changing

transient mean function, as exhibited by many queueing systems, still remains unresolved. The use of the point of maximum cusum deviation as a truncation point also needs additional analytical and empirical investigation. Combining information obtained from the cusum statistic with well-known heuristic truncation rules could prove to be useful in practice.

When K is large, the screening of alternatives by a generalized subset selection algorithm would be an effective means of reducing the required simulation run-lengths for all alternatives. This would be especially effective if for example the final subset of maximum size M were required to contain the L best of K covariance stationary processes. Ensuring that the L best alternatives are selected with probability P^* would reduce the field of candidates but still give the user flexibility in making the final selection.

As a refinement of the procedure developed in this research, guidelines for setting the initial sample size n_o should be formulated. During the experimental evaluation of the multiple ranking routines, it was observed that small initial sample sizes were consistently associated with over estimates of p(0) and hence with unnecessarily large run-lengths. Additionally, results obtained during this research have indicated the need to consider the ratio

$$\max \{[p_i(0)]^{1/2} : i = 1, \ldots, K\} \ / \ \Delta^*$$

when selecting the initial sample size. A well designed set of experiments could pinpoint relevant "rules of thumb" to assist the user in applying the MRP.

Finally, with the increasing application of simulation modelling in the nonacademic arena, additional effort to make the support routines developed in this research more "user friendly" would greatly expand the number of potential users.

APPENDIX A

```
C **  INTRODUCTION

C **  THIS PROGRAM CALCULATES A TEST STATISTIC FOR TESTING A COMPLETE
C **  SAMPLE FOR NORMALITY.  THE TEST STATISTIC IS OBTAINED BY DIVIDING
C **  THE SQUARE OF AN APPROPRIATE LINEAR COMBINATION OF THE SAMPLE ORDER
C **  STATISTICS BY THE USUAL SYMMETRIC ESTIMATE OF VARIANCE. THIS RATIO
C **  IS BOTH SCALE AND ORIGIN INVARIANT AND HENCE THE STATISTIC IS
C **  APPROPRIATE FOR A TEST OF THE COMPOSITE HYPOTHESIS OF NORMALITY


C **  THIS TEST WAS DEVELOPED BY S. S. SHAPIRO AND M. B. WILK
C **  A DESCRIPTION OF THE DEVELOPMENT APPEARS IN BIOMETRIKA (1965)
C **  VOL 52 PAGES 591-611.


C **  INCLUDED IN THIS PROGRAM IS A ROUTINE TO GENERATE AN ARMA(P,Q)
C **  PROCESS FOR USE AS A TEST SAMPLE.   PARAMETERS OF THE GENERATED
C **  TEST SEQUENCE MAY BE VARIED BY THE USER.


      PROGRAM WILK (TTY,OUTPUT,TAPE5=TTY,TAPE6=TTY,TAPE7,TAPE8,TAPE9)
C
C     SHAPIRO-WILK TEST FOR NORMALITY
C
      COMMON/BOX/OBS(32,50)
      COMMON /ABC/ N,J,K,SSQ,INDCOL(50),X(200),W
      DOUBLE PRECISION DSEED
      DATA INDCOL/0,0,0,1,3,5,8,11,15,19,24,29,35,41,48,55,63,71,80,89,9
     19,109,120,131,143,155,168,181,195,209,224,239,255,271,288,305,323,
     2341,360,379,399,419,440,461,483,505,528,551,575,599/
C
C     INITIALIZE VARIALBLES
C
      IPR = 0
      IROPT = 1
      NSIZE = 1
      DSEED = 12345.D0
      NREP = 16
      N = NREP
      KROW = 1
      NMAX = 3
      WRITE(9,150)


C **  BLOCK TO CALL SUBROUTINE WHICH GENERATES AN ARMA(P,Q)
C **  PROCESS TO TEST

      DO 1000 I = 1,NREP
        CALL ARMAPQ (DSEED,KROW)
        KROW = KROW + 1
        DSEED = DSEED*3.D0
 1000 CONTINUE
```

```
C **  DO LOOP TO BATCH OBSERVATIONS--STARTING WITH BATCH
C **  SIZE = 1.  LOOP WILL BE CALLED IF TEST FAILS FOR
C **  BATCH UNDER TEST WITH BATCH SIZE INCREASED BY 1

 1111 DO 10 I = 1,NREP
        SUM = 0.0
        DO 20 J = 1,NSIZE
          SUM = SUM + OBS(I,J)
          WRITE(9,*)I,J,OBS(I,J)
   20   CONTINUE
        AVG = SUM/NSIZE
        X(I) = AVG
   10 CONTINUE
C
C
C **  PRINT HEADER - NO. OF OBSERVATIONS
C
      WRITE (9,160) N
C
C
C **  SORT DATA POINTS IN ASCENDING ORDER
C
      CALL QSORT (X,N)
C
C **  CALCULATIONS
C
      IF (N.LE.50) J=INDCOL(N)
      K=N/2
      SUM=0.
      SSQ=0.
      DO 60 I=1,N
         SUM=SUM+X(I)
   60 SSQ=SSQ+X(I)*X(I)
      SSQ=SSQ-SUM*SUM/N
C
C **  PRINT ORDERED DATA IF NEEDED
C
      IF (IPR.NE.0) GO TO 70
      WRITE (9,180)
      WRITE (9,120) (X(I),I=1,N)
      WRITE (9,190)
C
C **  GET W STATISTIC **
C
   70 IF (N.LE.50) CALL TEST
C
C
C ** PERFORM SOME SPACING
C
      WRITE (9,210)
      IF (IPR.NE.0) WRITE (9,220)
      IF (IPR.EQ.0.AND.N.LE.30) WRITE (9,230)
      IF (IPR.EQ.0.AND.N.GT.30) WRITE (9,240)
```

```
C
C **   PRINT CRITICAL VALUES IF NEEDED
C
      CALL CRITVAL(N,CRIT)

C **  IF NORMALITY HYPOTHESIS IF REJECTED INCREASE
C **   BATCH SIZE (NSIZE)  BY 1 AND RETEST
C
C
      IF(W .LT. CRIT)THEN
      WRITE(9,190)
      WRITE(9,*)'NORMALITY REJECTED WITH NSIZE =',NSIZE
      WRITE(9,220)
      IF (NSIZE .GE. NMAX)THEN
          WRITE(9,*)'NSIZE TO LARGE TO USE THIS DATA SAMPLE'
          STOP
      END IF
          NSIZE = NSIZE + 1
          GO TO 1111
      END IF
      STOP
C
  120 FORMAT(5(5X,E14.4))
  150 FORMAT (1H1, 40H   SHAPIRO AND WILK W TEST FOR NORMALITY,/)
  160 FORMAT (26X,I3, 13H OBSERVATIONS,/)
  180 FORMAT (27X, 12HORDERED DATA,/)
  190 FORMAT (//)
  200 FORMAT (////////)
  210 FORMAT (1H )
  220 FORMAT (//////)
  230 FORMAT (///)
  240 FORMAT (//)
C
      END


      SUBROUTINE TEST
C
C **  THIS SUBROUTINE CALCULATES THE SHAPIRO-WILK W STATISTIC **
C
      COMMON /ABC/ N,J,K,SSQ,INDCOL(50),X(200),W
      COMMON /COEF/ A(625)
C
      B=0
      DO 10 I=1,K
   10 B=B+A(J+I)*(X(N-I+1)-X(I))
      W=B*B/SSQ
      WRITE (9,20) W
      RETURN
C
   20 FORMAT (19X, 17HSHAPIRO-WILK W = ,F6.4)
C
      END
```

```
      SUBROUTINE CRITVAL (N,CRIT)
C
C **  PRINT THE CRITICAL VALUES FOR THE W TEST
C
      COMMON /CRIT/ T(50,1)
C
      WRITE (9,50)
      WRITE (9,60) N,T(N,1)
      WRITE (9,90)
      CRIT = T(N,1)
      RETURN
C
   50 FORMAT (19X, 28HCRITICAL VALUE OF THE W-TEST)
   60 FORMAT (/12X, 41HN    0.10                                   ,//I13,
     1F8.3//)
   90 FORMAT (//16X, 44HNOTE THAT SMALL VALUES OF W ARE SIGNIFICANT,,/16
     1X, 48HI.E., LEAD TO REJECTION OF THE NORMAL HYPOTHESIS)
C
      END
      BLOCK DATA W
      COMMON /COEF/ A(200),C(200),D(200),F(25)
      COMMON /CRIT/ T(50,1)
      DATA A/.7071,.6872,.1677,.6646,.2413,.6431,.2806,.0875,.6233,.3031
     1,.1401,.6052,.3164,.1743,.0561,.5888,.3244,.1976,.0947,.5739,.3291
     2,.2141,.1224,.0399,.5601,.3315,.2260,.1429,.0695,.5475,.3325,.2347
     3,.1586,.0922,.0303,.5359,.3325,.2412,.1707,.1099,.0539,.5251,.3318
     4,.2460,.1802,.1240,.0727,.0240,.5150,.3306,.2495,.1878,.1353,.0880
     5,.0433,.5056,.3290,.2521,.1939,.1447,.1005,.0593,.0196,.4968,.3273
     6,.2540,.1988,.1524,.1109,.0725,.0359,.4886,.3253,.2553,.2027,.1587
     7,.1197,.0837,.0496,.0163,.4808,.3232,.2561,.2059,.1641,.1271,.0932
     8,.0612,.0303,.4734,.3211,.2565,.2085,.1686,.1334,.1013,.0711,.0422
     9,.0140,.4643,.3185,.2578,.2119,.1736,.1399,.1092,.0804,.0530,.0263
     *,.4590,.3156,.2571,.2131,.1764,.1443,.1150,.0878,.0618,.0368,.0122
     *,.4542,.3126,.2563,.2139,.1787,.1480,.1201,.0941,.0696,.0459,.0228
     *,.4493,.3098,.2554,.2145,.1807,.1512,.1245,.0997,.0764,.0539,.0321
     *,.0107,.4450,.3069,.2543,.2148,.1822,.1539,.1283,.1046,.0823,.0610
     *,.0403,.0200,.4407,.3043,.2533,.2151,.1836,.1563,.1316,.1089,.0876
     *,.0672,.0476,.0284,.0094,.4366,.3018,.2522,.2152,.1848,.1584,.1346
     *,.1128,.0923,.0728,.0540,.0358,.0178,.4328,.2992,.2510,.2151,.1857
     *,.1601,.1372,.1162,.0965,.0778,.0598,.0424,.0253,.0084,.4291,.2968
     *,.2499,.2150/
      DATA C/.1864,.1616,.1395,.1192,.1002,.0822,.065,.0483,.032,.0159,.
     14254,.2944,.2487,.2148,.1870,.1630,.1415,.1219,.1036,.0862,.0697,.
     20537,.0381,.0227,.0076,.4220,.2921,.2475,.2145,.1874,.1641,.1433,.
     31243,.1066,.0899,.0739,.0585,.0435,.0289,.0144,.4188,.2898,.2463,.
     42141,.1878,.1651,.1449,.1265,.1093,.0931,.0777,.0629,.0485,.0344,.
     50206,.0068,.4156,.2876,.2451,.2137,.1880,.1660,.1463,.1284,.1118,.
     60961,.0812,.0669,.0530,.0395,.0262,.0131,.4127,.2854,.2439,.2132,.
     71882,.1667,.1475,.1301,.1140,.0988,.0844,.0706,.0572,.0441,.0314,.
     80187,.0062,.4096,.2834,.2427,.2127,.1883,.1673,.1487,.1317,.1160,.
     91013,.0873,.0739,.0610,.0484,.0361,.0239,.0119,.4068,.2813,.2415,.
     *2121,.1883,.1678,.1496,.1331,.1179,.1036,.0900,.0770,.0645,.0523,.
```

```
     *0404,.0287,.0172,.0057,.4040,.2794,.2403,.2116,.1883,.1683,.1505,.
     *1344,.1196,.1056,.0924,.0798,.0677,.0559,.0444,.0331,.0220,.0110,.
     *4015,.2774,.2391,.2110,.1881,.1686,.1513,.1356,.1211,.1075,.0947,.
     *0824,.0706,.0592,.0481,.0372,.0264,.0158,.0053,.3989,.2755,.2380,.
     *2104,.1880,.1689,.1520,.1366,.1225,.1092,.0967,.0848,.0733,.0622,.
     *0515,.0409,.0305,.0203,.0101,.3964,.2737,.2368,.2098,.1878,.1691,.
     *1526,.1376,.1237,.1108,.0986,.0870,.0759,.0651,.0546,.0444,.0343,.
     *0244,.0146,.0049/
      DATA D/.394,.2719,.2357,.2091,.1876,.1693,.1531,.1384,.1249,.1123,
     1.1004,.0891,.0782,.0677,.0575,.0476,.0379,.0283,.0188,.0094,.3917,
     2.2701,.2345,.2085,.1874,.1694,.1535,.1392,.1259,.1136,.1020,.0909,
     3.0804,.0701,.0602,.0506,.0411,.0318,.0227,.0136,.0045,.3894,.2684,
     4.2334,.2078,.1871,.1695,.1539,.1398,.1269,.1149,.1035,.0927,.0824,
     5.0724,.0628,.0534,.0442,.0352,.0263,.0175,.0087,.3872,.2667,.2323,
     6.2072,.1868,.1695,.1542,.1405,.1278,.1160,.1049,.0943,.0842,.0745,
     7.0651,.0560,.0471,.0383,.0296,.0211,.0126,.0042,.3850,.2651,.2313,
     8.2065,.1865,.1695,.1545,.1410,.1286,.1170,.1062,.0959,.0860,.0765,
     9.0673,.0584,.0497,.0412,.0328,.0245,.0163,.0081,.3830,.2635,.2302,
     *.2058,.1862,.1695,.1548,.1415,.1293,.1180,.1073,.0972,.0876,.0783,
     *.0694,.0607,.0522,.0439,.0357,.0277,.0197,.0118,.0039,.3808,.2620,
     *.2291,.2052,.1859,.1695,.1550,.1420,.1300,.1189,.1085,.0986,.0892,
     *.0801,.0713,.0628,.0546,.0465,.0385,.0307,.0229,.0153,.0076,.3789,
     *.2604,.2281,.2045,.1855,.1693,.1551,.1423,.1306,.1197,.1095,.0998,
     *.0906,.0817,.0731,.0648,.0568,.0489,.0411,.0335,.0259,.0185,.0111,
     *.0037,.3770,.2589,.2271,.2038,.1851,.1692,.1553,.1427,.1312,.1205,
     *.1105,.1010,.0919,.0832,.0748,.0667,.0588,.0511,.0436,.0361,.0288,
     *.0215,.0143,.0071/
      DATA F/.3751,.2574,.226,.2032,.1847,.1691,.1554,.143,.1317,.1212,.
     11113,.1020,.0932,.0846,.0764,.0685,.0608,.0532,.0459,.0386,.0314,.
     20244,.0174,.0104,.0035/
       DATA T/0.,0.,.789,.792,.806,.826,.838,.851,.859,.869,.8
     *76,.883,.889,.895,.901,.906,.910,.914,.917,.920,.923,.926,.928,.93
     *0,.931,.933,.935,.936,.937,.939,.940,.941,.942,.943,.944,.945,.946
     *,.947,.948,.949,.950,.951,.951,.952,.953,.953,.954,.954,.955,.955/
C
      END
      SUBROUTINE QSORT (X,N)
C
C     QUICKSORT ALGORITHM.
C
      DIMENSION X(1), STACK(13,2)
      INTEGER STACK,FIRST
      REAL MEDIAN,MED
      DATA MAXSTK/13/,K/12/,M/10/
      ITOP=0
      FIRST=1
      NN=N
   10 CONTINUE
      IF (NN.GT.M) GO TO 20
      CALL SHLSRT (X(FIRST),NN)
      IF (ITOP.LE.0) GO TO 130
      FIRST=STACK(ITOP,1)
      NN=STACK(ITOP,2)
      ITOP=ITOP-1
      GO TO 10
```

```
   20 CONTINUE
      LAST=FIRST+NN-1
      N1=0
      N2=0
      MEDIAN=MED(X(FIRST),NN)
      I1=FIRST
      I2=LAST+1
   30 CONTINUE
      I=I2-1
   40 CONTINUE
      IF (I.LE.I1) GO TO 80
      IF (X(I).LT.MEDIAN) GO TO 50
      I=I-1
      N2=N2+1
      GO TO 40
   50 I2=I
      X(I1)=X(I2)
      N1=N1+1
      I=I1+1
   60 CONTINUE
      IF (I.GE.I2) GO TO 90
      IF (X(I).GT.MEDIAN) GO TO 70
      I=I+1
      N1=N1+1
      GO TO 60
   70 I1=I
      X(I2)=X(I1)
      N2=N2+1
      GO TO 30
   80 X(I1)=MEDIAN
      GO TO 100
   90 X(I2)=MEDIAN
  100 CONTINUE
      ITOP=ITOP+1
      IF (ITOP.GT.MAXSTK) GO TO 120
      IF (N1.GT.N2) GO TO 110
      STACK(ITOP,1)=LAST-N2+1
      STACK(ITOP,2)=N2
      NN=N1
      GO TO 10
  110 STACK(ITOP,1)=FIRST
      STACK(ITOP,2)=N1
      FIRST=LAST-N2+1
      NN=N2
      GO TO 10
  120 CALL REMARK (24LSTACK OVERFLOW IN QSORT )
      STOP
  130 RETURN
C
      END
      SUBROUTINE SHLSRT (X,N)
C
C     SHELL SORT.
C
      DIMENSION X(1)
```

```
      M=N
   10 CONTINUE
      M=M/2
      IF (M.EQ.0) GO TO 50
      K=N-M
      J=1
   20 CONTINUE
      I=J
   30 CONTINUE
      L=M+I
      IF (X(I).LE.X(L)) GO TO 40
      XK=X(I)
      X(I)=X(L)
      X(L)=XK
      I=I-M
      IF (I.GE.1) GO TO 30
   40 CONTINUE
      J=J+1
      IF (J-K) 20,20,10
   50 CONTINUE
      RETURN
C
      END
      FUNCTION MED (X,N)
C
C     FUNCTION TO GET A MEDIAN ESTIMATE OF AN ARRAY.
C
      REAL MED
      DIMENSION X(1)
      MID=N/2
      XF=X(1)
      XM=X(MID)
      XL=X(N)
      IF (XF.GT.XM) GO TO 10
      IF (XM.LT.XL) GO TO 30
      IF (XF.LT.XL) GO TO 40
      GO TO 20
   10 IF (XM.GT.XL) GO TO 30
      IF (XF.GT.XL) GO TO 40
   20 K=1
      GO TO 50
   30 K=MID
      GO TO 50
   40 K=N
   50 MED=X(K)
      X(K)=X(1)
      RETURN
C
      END


C **  SUBROUTINE TO GENERATE ARMA(P,Q) PROCESS FOR TEST
```

```
      SUBROUTINE ARMAPQ(SEED,KROW)
      COMMON/BOX/TEST(32,50)
      COMMON/TSERIES/DELTA(3),SIGMA(3),NSAMP(3),IP(3),JQ(3),
     1THETA(3,48),X(3,60),U(3,60),IOLD(3),JOLD(3),PHI(3,48),
     2ONE(1),DSEED
      DOUBLE PRECISION DSEED,SEED
C*****   INITIALIZE
      DSEED = SEED
      DELTA(1) = 110.0
      SIGMA(1) = 20.0
      IP(1) = 2
      JQ(1) = 0
      PHI(1,1) = 0.35
      PHI(1,2) = 0.25
      PHI(1,3) = 0.0
      THETA(1,1) = 0.0
      THETA(1,2) = 0.0
      THETA(1,3) = 0.0
      NR = 1
      START = ARMA(0,1)
      DO 100 I=1,1000
      CALL GGNML(DSEED,NR,ONE)
        CLEAR = ARMA(1,1)
  100 CONTINUE
      DO 200 I=1,50
      CALL GGNML(DSEED,NR,ONE)
        TEST(KROW,I) = ARMA(1,1)
  200 CONTINUE
      RETURN
      END
      FUNCTION ARMA(IND,KS)
C *** GENERATE ARMA (P,Q) MODELS
C *** GENERATOR USES ARRAYS,X(SERIES) & U(WHITE NOISE SERIES),
C     TO ACCOUNT FOR DEPENDENT PAST VALUES. IOLD AND JOLD POINT TO
C      THE OLDEST ELEMENT IN EACH ARRAY.  NEWEST ELEMENT IS ONE
C     ELEMENT OVER.
      COMMON/TSERIES/DELTA(3),SIGMA(3),NSAMP(3),IP(3),JQ(3),
     1THETA(3,48),X(3,60),U(3,60),IOLD(3),JOLD(3),PHI(3,48)
     2,ONE(1),DSEED
      DOUBLE PRECISION DSEED
C *** FIRST TIME THROUGH (IND=0), INITILIZE VARIABLES. OTHERWISE,
C     GO TO 100 AND GENERATE SERIES.
      IF(IND.EQ.1) GO TO 100
      NIP=IP(KS)
      NJQ=JQ(KS)
      XMU = DELTA(KS)
      SUM = 1.0
C *** CALCULATE MAXIMUM LAG, LMAX
      LMAX = MAX0(NIP,NJQ)
C *** CALCULATE MEAN (XMU) OF SERIES
      IF (NIP .EQ. 0) GO TO 20
      DO 10 I=1,NIP
   10 SUM = SUM - PHI(KS,I)
   20 XMU = DELTA(KS)/SUM
C *** INITIALIZE OLDEST ELEMENT POINTERS, IOLD & JOLD, FOR SERIES (X)
```

```
C     & WHITE NOISE SERIES (U) TO LAST ELEMENT IN EACH ARRAY
      IOLD(KS) = NIP
      JOLD(KS) = NJQ
      DO 30 LAG=1,LMAX
C *** INITIALIZE WHITE NOISE SERIES TO MEAN (0.)
      U(KS,LAG) = 0.0
C *** INITIALIZE SERIES (X,ARMA) TO MEAN (XMU)
   30 X(KS,LAG) = XMU
   35 ARMA = XMU
      RETURN
C *** WHITE NOISE (UO) IS NORMAL(0.,SIGMA)
  100 CALL GGNML(DSEED,NR,ONE)
      UO = SIGMA(KS)*ONE(KS)
      ARMA = DELTA(KS) + UO
C *** IF ARMA NOT DEPENDENT ON PAST SERIES VALUES (X), DON'T
C     ADD THEM ON
C *** ARMA DEPENDS ON WHITE NOISE PLUS DELTA TO BRING SERIES
C     UP TO MEAN
      IF (IP(KS) .EQ. 0) GO TO 150
C *** GET PAST SERIES ELEMENTS (X) IN ORDER, FROM LAST TO
C     OLDEST
      DO 120 II=1,IP(KS)
      I = MOD(IOLD(KS)+II,IP(KS))
      IF (I.EQ.0) I=IP(KS)
C *** ADD TO ARMA PAST SERIES VALUES(X) TIMES PHI ARRAY
  120 ARMA = ARMA + PHI(KS,II)*X(KS,I)
C *** IF ARMA NOT DEPENDENT ON PAST WHITE NOISE VALUES(U),
C     DON'T ADD THEM ON
  150 IF (JQ(KS) .EQ. 0) GO TO 500
C *** GET PAST WHITE NOISE VARIABLES (U) FROM LAST PERIOD
C     TO OLDEST
      DO 170 JJ=1,JQ(KS)
      J = MOD(JOLD(KS)+JJ,JQ(KS))
      IF (J.EQ.0) J=JQ(KS)
C *** SUBTRACT PAST WHITE NOISE VARIABLES(U) TIMES THETA ARRAY
  170 ARMA = ARMA - THETA(KS,JJ)*U(KS,J)
C *** IF ARMA IS DEPENDENT ON PAST SERIES VALUES (X), SAVE
C     ARMA WHERE OLDEST X ELEMENT IS.
  500 IF (IP(KS) .EQ. 0) GO TO 550
      X(KS,IOLD(KS)) = ARMA
C *** UPDATE IOLD WHERE IOLD IS BETWEEN 1 AND P
      IOLD(KS) = IOLD(KS) - 1
      IF (IOLD(KS) .EQ. 0) IOLD(KS) = IP(KS)
C *** IF ARMA NOT DEPENDENT ON PAST WHITE NOISE, DON'T UPDATE
C     U ARRAY
  550 IF (JQ(KS) .EQ. 0) RETURN
C *** SAVE CURRENT WHITE NOISE (UO) WHERE OLDEST WHITE NOISE
C     HAD BEEN
      U(KS,JOLD(KS)) = UO
C *** UPDATE JOLD
      JOLD(KS) = JOLD(KS) - 1
      IF (JOLD(KS) .EQ. 0) JOLD(KS) = JQ(KS)
      RETURN
      END
```

APPENDIX B

```
C **  PROGRAM IBZERO IS USED TO DETECT THE PRESENCE TO INITIALIZATION
C **  BIAS AND AUTOMATICALLY TRUNCATE THE DATA UNDER TEST.
C **  THE DATA TO BE TESTED SHOULD BE NAMED TAPE8.  IT WILL BE READ
C **  AFTER THE USER ANSWERS QUESTIONS PROMPTED ON THE CRT.

C **  THE DATA TO BE TESTED MAY BE BATCHED PRIOR TO ANY TEST BEING
C **  CONDUCTED BY CHANGING THE DATA ELEMENT -NSIZE- NSIZE IS PRESENTLY
C **  SET AT 1 (NO BATCHING).

C **  THIS PROGRAM IS A MODIFICATION OF A TESTED DESIGNED BY L. SCRUBEN
C **  AS REPORTED IN VOL 30 MAY 1982 OPERATIONS RESEARCH.

C **  THE REQUIRED ESTIMATE OF THE DATA VARIANCE IS MEASURED BY
C **  USING THE SPECTRAL METHOD

C **  INCLUDED IN THE PROGRAM ARE TO PLOT SUBROUTINES
C **  THESE ARE USED TO PLOT THE BATCH MEANS AND THE CUSUMS

      PROGRAM IBZERO(TTY,OUTPUT,TAPE5=TTY,TAPE6=TTY,TAPE7,TAPE8,TAPE9)
      DIMENSION DATA(2000)
      COMMON//PLT(2000),NUMBAT,BATCH(2000)
      DATA NSIZE/1/
C
C
C
C   *****REQUEST USER INPUT FROM TERMINAL
C   *****NUMBER OF DATA POINTS IN FILE(TAPE8)
C
      AGAIN = 0.0
      WRITE(6,*)'ENTER NUMBER OF DATA POINTS'
      READ(5,*)OBS
      LENTH = IFIX(OBS)
C
C   *****REQUEST USER INPUT FROM TERMINAL
C   *****THE PRESPECIFIED LEVEL OF SIGNIFICANCE
C
   60 WRITE(6,*)'ALPHA = '
      READ(5,*)ALPHA
C
C   *****CHECK THAT USER INPUT PROPER DATA TYPE
C
      IF((ALPHA .LE. 0.) .OR.(ALPHA .GE. 1.0))THEN
          WRITE(6,52)
   52     FORMAT(/,'YOUR VALUE FOR ALPHA IS INCORRECT.',
     +    ' PLEASE REENTER.')
          GO TO 60
      END IF
C
C   *****INITIALIZE VARIABLES
C
      OBS = LENTH
      JUNK = 0
```

```
      IBEGIN = 1
      MPOINT = 0
      IEND = NSIZE
      ITRUNC = 0
      KOUNT = 0
C
C
C
C  *****DO LOOP TO LOAD DATA  INTO ARRAY FOR PROCESSING
C
      IF(AGAIN .NE. 0.0)THEN
          GO TO 75
      END IF
      READ(8,*,ERR=901) (DATA(I),I=1,LENTH)
C
C
   75 NUMBAT = OBS/NSIZE
      NSTART = IBEGIN
      POINTS = FLOAT(NUMBAT)
      KOUNT = KOUNT + 1
      NFIN = IEND
      DO 150 K=1,NUMBAT
      BATSUM = 0.0
      DO 100 I=NSTART,NFIN
      BATSUM = BATSUM + DATA(I)
  100 CONTINUE
C
C  *****COMPUTE MEAN OF EACH BATCH
C
      BATCH(K) = BATSUM/NSIZE
      NSTART = NSTART + NSIZE
      NFIN = NFIN + NSIZE
  150 CONTINUE
C
C
C  *****CALL SUBROUTINE TO PLOT BATCH MEANS
C
      IF(KOUNT .EQ. 1)THEN
          WRITE(9,*)'FOLLOWING IS A PLOT OF BATCH MEANS'
          CALL PBATCH
      END IF
C
C  *****COMPUTE SUM OF SAFE BATCH MEANS
C
      TOT = 0.0
      NHALF = NUMBAT/2
      MID = NHALF + 1
      NSAFE = NUMBAT - NHALF
C
C
C#####  CALL SUBROUTINE WELCH TO CALCULATE VARIANCE  ##################
C
C
      CALL WELCH(MID,ZERO)
      WRITE(9,*)'ZERO =',ZERO
```

```
C
C
C
      DO 200 I=MID,NUMBAT
        TOT = TOT + BATCH(I)
  200 CONTINUE
C
C
C
      GAMMA = ZERO
C
C  *****INITIALIZE VARIABLES USED DURING
C  *****EACH PASS THROUGH THE DATA
C
      AMAX = 0.0
      CUSUM = 0.0
      PMEAN = 0.0
      AMIN = 0.0
      PSUM = 0.0
      M = 0
      NEGTIV = 0
      POSTIV = 0
      TOTAL = 0.0
C
C  *****COMPUTE MEAN OF ALL DATA
C
      DO 325 I = 1,NUMBAT
        TOTAL = TOTAL + BATCH(I)
  325 CONTINUE
      AMEAN = TOTAL/NUMBAT
      IF(KOUNT .EQ. 1)TMEAN=AMEAN
      WRITE(9,326)KOUNT,AMEAN,TMEAN
  326 FORMAT(/,5X,'PASS NUMBER',I5,' MEAN=',F10.4,' TMEAN=',F10.4)
C
C
C  *****TEST FOR INITIALIZATION BIAS
C  *****FIND MOST POSITIVE AND NEGATIVE
C  *****VALUES OF NORMALIZED CUSUM
C
      SIGMA = SQRT(GAMMA)
      SQROOT = SQRT(POINTS)
      DO 500 I=1,NUMBAT
        M = M+1
        PSUM = PSUM+BATCH(I)
        PMEAN = PSUM/M
        CUSUM = AMEAN-PMEAN
C
C  *****BLOCK TO CHECK FOR NEGATIVE VALUES OF CUSUM
C  *****AND SAVE MOST NEGATIVE VALUE
C
      IF(CUSUM .LT. 0.0)THEN
          NEGTIV = 1
          SNEG = (M*CUSUM)/SQROOT
          PLT(I) = SNEG/SIGMA
          IF(SNEG.LT.AMIN)THEN
```

```
              AMIN=SNEG
              NEGLOC = M
          END IF
          GO TO 500
      END IF
C
C
        STAR = (M*CUSUM)/SQROOT
      PLT(I) = STAR/SIGMA
        IF(STAR .GT. AMAX)THEN
            POSTIV = 1
            AMAX = STAR
            MPOINT = M
        END IF
  500 CONTINUE
C
C  *****PLOT S VALUES******
C
      WRITE(9,*)'FOLLOWING IS A PLOT OF STAR DURING PASS',KOUNT
      CALL PSTAR
C  *****BLOCK TO CHECK IF ONLY POSITIVE
C  *****INITIAL BIAS INDICATED
C
      IF((NEGTIV .GT. 0) .AND.
     +(POSTIV .LT. 1.0))THEN
      WRITE(9,*)'INDICATIONS OF POSITIVE BIAS ONLY DURING PASS',KOUNT
          AMAX = AMIN
          MPOINT =NEGLOC
          GO TO 501
      END IF
C
C  *****BLOCK TO CHECK IF OSCILLATION OF
C  *****NEGATIVE AND POSITIVE BIAS INDICATED.
C
C  *****STANDARDIZE TO UNIT INTERVAL.
C
C  *****CHECK USING SAME SCHRUBEN TEST
C  *****EXCEPT USING ALPHA/2.
C
      IF((NEGTIV .GT. 0).AND.(POSTIV.GT.0))THEN
          WRITE(9,*)'INDICATION OF + AND - BIAS DURING PASS',KOUNT
          TN = FLOAT(NEGLOC)/POINTS
          TP = FLOAT(MPOINT)/POINTS
          XP=(AMAX**2)/(3*GAMMA*TP*(1-TP))
          XN=(AMIN**2)/(3*GAMMA*TN*(1-TN))
          DFN =3.
          DFD=POINTS/2
          X = XP
          CALL MDFDRE(X,DFN,DFD,P,IER)
          PROPOS = 1. - P
          X =XN
          CALL MDFDRE(X,DFN,DFD,P,IER)
          PRONEG = 1.0 - P
          HALPHA = ALPHA/2.
          IF((PROPOS .AND.PRONEG)
```

```
     +      .LT. HALPHA)THEN
                 MPOINT=MAXO(MPOINT,NEGLOC)
                 GO TO 913
             END IF
             IF((PROPOS .AND. PRONEG)
     +      .GE. HALPHA)THEN
                 GO TO 974
             END IF
         IF(PROPOS .LT. HALPHA)THEN
                 GO TO 913
             ELSE
                 MPOINT = NEGLOC
                 GO TO 913
             END IF
         END IF
C
C  *****BLOCK TO CALCULATE VIA SCHRUBEN
C  *****BROWNIAN BRIDGE TEST INDICATION
C  *****OF INITIAL BIAS OF ONLY ONE SIGN
C
      WRITE(9,*)'INDICATIONS OF ONLY - BIAS DURING PASS ',KOUNT
  501 T = FLOAT(MPOINT)/POINTS
      X = (AMAX**2)/(3.*GAMMA*T*(1.-T))
      DFN = 3.
      DFD = POINTS/2
      CALL MDFDRE(X,DFN,DFD,P,IER)
      IF(IER .EQ. 129)THEN
          WRITE(9,*)'IER ERROR'
          STOP
      END IF
      PROBAB = 1.0 - P
      IF(PROBAB .LT. ALPHA)THEN
C
C  *****BLOCK TO OVERRIDE,IF NECESSARY,
C  *****TRUCATION POINT TO ALLOW AT LEAST TWO
C  *****PASSES TO ELIMINATE INITIAL BIAS
C  *****POINT = .25*DATA
C
C
  913     CONTINUE
          MAXPNT = IFIX(.25*NUMBAT)
          IF(MPOINT .GT. MAXPNT)THEN
              MPOINT = MAXPNT
          END IF
C
C
C  *****ITRUNC EQUALS THE TRUNCATION POINT OF BATCHES
C
      ITRUNC = ITRUNC + MPOINT
C
C  *****BLOCK TO SEE IF THE TEST PROCEDURE HAS
C  *****TRUNCATED AN EXCESSIVE AMOUNT OF DATA
C  *****(50%) AND STILL NOT ELIMINATED INITIAL BIAS
C
      JUNK = ITRUNC*NSIZE
```

```
      WRITE(9,924)KOUNT,JUNK
  924 FORMAT(//,'PASS NUMBER',I3,
     +' THROUGH THE DATA SHOWS A TRUNCATION POINT OF ',I4)
C
      IF(JUNK .GT. (0.5*LENTH))THEN
          WRITE(9,925)JUNK
  925 FORMAT(/,'AN EXCESSIVE AMOUNT OF DATA(AT LEAST',I3,
     +' DATA POINTS) MUST BE TRUNCATED',/,' TO ELIMINATE ',
     +'INITIALIZATION BIAS. IT IS SUGGESTED THAT A',/,
     +' LARGER SAMPLE BE OBTAINED AND THE TEST RERUN')
      GO TO 929
      END IF
C
C  *****CALCULATE NUMBER OF DATA POINTS
C  *****LEFT AND RETEST USING ONLY THESE POINTS
C
          OBS = OBS - (MPOINT*NSIZE)
          IBEGIN = IBEGIN + (MPOINT*NSIZE)
          IEND = IBEGIN + (NSIZE - 1)
          GO TO 75
      ELSE
C
C
C
C  *****SHOW FINAL TRUNCATION POINT
C  *****AND COMPUTED MEAN
C
  974     WRITE(9,975)ALPHA,JUNK
  975     FORMAT(///,'THE HYPOTHESIS OF NO  INITIALIZATION',
     +    ' BIAS IS NOT REJECTED AT THE PRESPECIFIED',/,
     +    ' LEVEL OF SIGNIFICANCE ',F5.4,' WITH A TRUNCATION',
     +    ' POINT OF ',I4)
      END IF
      WRITE(9,*)'MEAN OF RAW DATA = ',TMEAN
      WRITE(9,981)AMEAN
  981 FORMAT(/,'THE ARITHMETIC MEAN OF THE',
     +' TRUNCATED DATA = ',F10.4)
      AGAIN = 0.0
C
C  *****ASK IF RERUN WANTED WITH DIFFERENT ALPHA LEVEL
C
  929 WRITE(6,930)
  930 FORMAT(//,'DO YOU WANT TO RERUN THE DATA WITH A ',
     +'DIFFERENT LEVEL OF SIGNIFICANCE?',/,
     +'ENTER 0 FOR NO',/,'ENTER 1 FOR YES')
      READ(6,*)AGAIN
      IF(AGAIN .GT. 0.0)THEN
          GO TO 60
      END IF
      STOP
C
C  *****ERORR MESSAGE
C
  901 WRITE(6,*)'THERE IS A  DATA ERROR IN YOUR FILE. PLEASE RECHECK'
      STOP
```

```
      END
C
C
C
C
C          SUBROUTINE PLOT
C
C
      SUBROUTINE PSTAR
      COMMON//PLT(2000),NUMBAT,BATCH(2000)
      WRITE(9,201)
  201 FORMAT(//////)
      YBIG=PLT(1)
      YMIN=PLT(1)
      DO 100 I=2,NUMBAT
      IF(YMIN.LT.PLT(I))GO TO 50
      YMIN=PLT(I)
   50 CONTINUE
      IF(YBIG.GT.PLT(I) )GO TO 100
      YBIG=PLT(I)
  100 CONTINUE
      WRITE(9,350)(1H*,J=1,55)
      DO 200 I=1,NUMBAT
      IF(YBIG.EQ.YMIN)THEN
      L=IFIX(49.999*PLT(I) )+2
      ELSE
      L=IFIX(49.999*(PLT(I) -YMIN)/(YBIG-YMIN))+2
      END IF
  350 FORMAT(22X,60A1)
      IF(PLT(I).GT.0.0)THEN
      WRITE(9,300)I,PLT(I),(1H ,J=1,L-1),1H+,(1H ,J=1,53-L)
      GO TO 200
      END IF
      IF(PLT(I).LT.0.0)THEN
      WRITE(9,300)I,PLT(I),(1H ,J=1,L-1),1H-,(1H ,J=1,53-L)
      GO TO 200
      END IF
      WRITE(9,300)I,PLT(I),(1H ,J=1,L-1),1H0,(1H ,J=1,53-L)
  200 CONTINUE
  300 FORMAT(1H ,I5,1H ,F8.4,2X,1H*,60A1)
      WRITE(9,350)(1H*,J=1,54)
      RETURN
      END
C
C
C     SUBROUTINE TO PLOT BATCH AND PLT
C
C
      SUBROUTINE PBATCH
      COMMON//PLT(2000),NUMBAT,BATCH(2000)
      WRITE(9,201)
  201 FORMAT(//////)
      YBIG=BATCH(1)
      YMIN=BATCH(1)
      DO 100 I=2,NUMBAT
```

```
      IF(YMIN.LT.BATCH(I))GO TO 50
      YMIN=BATCH(I)
   50 CONTINUE
      IF(YBIG.GT.BATCH(I) )GO TO 100
      YBIG=BATCH(I)
  100 CONTINUE
      WRITE(9,350)(1H*,J=1,55)
      DO 200  I=1,NUMBAT
      IF(YBIG.EQ.YMIN)THEN
      L=IFIX(49.999*BATCH(I) )+2
      ELSE
      L=IFIX(49.999*(BATCH(I) -YMIN)/(YBIG-YMIN))+2
      END IF
  350 FORMAT(22X,60A1)
      IF(BATCH(I).GT.0.0)THEN
      WRITE(9,300)I,BATCH(I),(1H ,J=1,L-1),1H+,(1H ,J=1,53-L)
      GO TO 200
      END IF
      IF(BATCH(I).LT.0.0)THEN
      WRITE(9,300)I,BATCH(I),(1H ,J=1,L-1),1H-,(1H ,J=1,53-L)
      GO TO 200
      END IF
      WRITE(9,300)I,BATCH(I),(1H ,J=1,L-1),1H0,(1H ,J=1,53-L)
  200 CONTINUE
  300 FORMAT(1H ,I5,1H ,F8.4,2X,1H*,60A1)
      WRITE(9,350)(1H*,J=1,54)
      RETURN
      END
C#####   SUBROUTINE TO CALCULATE THE SPECTRAL DENSITY   ################
C#####   AT ZERO FREQUENCY (VARIANCE)                   #################
C
C
      SUBROUTINE WELCH (MID,ZERO)
      DIMENSION PERIOD(600),XM(6),TEMP(6),B(6,7)
     $,ANOVA(16),VARB(21),FJ(300),IWK(3200),WK(3200),V(300,6),VCV(21)
     $,NBR(6),ALFA(2),IJOB(2),IND(11),XYB(6,5),A(300,6),CHECK(2000)
      COMMON//PLT(2000),NUMBAT,BATCH(2000)
      COMPLEX TRANS(600)
C
C
      NUMDAT = NUMBAT - MID + 1
      INUM = 0
      DO 10 I = MID,NUMBAT
        INUM = INUM + 1
        CHECK(INUM) = BATCH(I)
   10 CONTINUE
      NHDATA = NUMDAT/2
      NQDATA = NHDATA/2
      CALL FFTRC(CHECK,NUMDAT,TRANS,IWK,WK)
      MMM = NHDATA + 1
      DO 15 L=2,MMM
        K = L - 1
        PERIOD(K)=(CABS(TRANS(L))**2)/NUMDAT
   15 CONTINUE
      DO 30 KL=1,NQDATA
```

```
        KK =2*KL
        JJ = KK-1
      SMOTH=((PERIOD(JJ)+PERIOD(KK))/2.0)
      FJ(KL)=ALOG(SMOTH) + .270
   30 CONTINUE
      DO 50 I=1,NQDATA
          A(I,1) =   I
          A(I,2) =   I*I
          A(I,3) = I**3
          A(I,4) = I**4
          A(I,5) = I**5
          A(I,6) = FJ(I)
   50 CONTINUE
      M = 5
      IB = 6
      IND(1) = 0
      IND(2) = 0
      IND(3) = 0
      IND(4) = 0
      IND(5) = 0
      IJOB(1) = 0
      IJOB(2) = 1
      ALFA(1) = .05
      ALFA(2) = .05
      NRDIM = 300
      CALL RLSEP(A,NQDATA,M,NRDIM,ALFA,IJOB,IND,ANOVA,XYB,IB,VARB,IER)
      J = 0
      DO 2000 I=1,5
        IF (XYB(I,2) .NE. 0)THEN
            J = J + 1
            DO 2100 LL = 1,NQDATA
              V(LL,J) = LL**I
 2100       CONTINUE
        END IF
 2000 CONTINUE
      IF(J .EQ. 0)THEN
          ZERO = EXP(XYB(6,2))
          RETURN
      ELSE
          J = J + 1
          DO 2200 I = 1,NQDATA
            V(I,J) = FJ(I)
 2200     CONTINUE
      NVAR = J
      NBR(1) = NVAR
      NBR(2) = NQDATA
      NBR(3) = NQDATA
      NBR(4) = 1
      NBR(5) = 1
      NBR(6) = 1
      CALL BECOVM(V,NRDIM,NBR,TEMP,XM,VCV,IER)
      IVAR = NVAR - 1
      ALPHA = 0.05
      CALL RLMUL(VCV,XM,NQDATA,IVAR,ALPHA,ANOVA,B,IB,VARB,IER)
      UPLEFT = (B(NVAR,4)**2)/ANOVA(8)
```

```
ADJUST = (.645*UPLEFT)/2.0
CONE = EXP(-ADJUST)
ZERO = CONE*(EXP(B(NVAR,1)))
END IF
RETURN
END
```

APPENDIX C

```
C **   INTRODUCTION


C **   PROGRAM TO PERFORM MULTIPLE REPLICATIONS OF
C **   DUDEWICZ-DALAL MRP USING INDEPENDENT NORMALLY
C **   GENERATED SAMPLES.  TEST SAMPLES GENERATED ARE
C **   IN A LEAST FAVORABLE CONFIGURATION.
C **   NUMBER OF ALTERNATIVE(KPOP WITH MAX 9) CONSIDERED,
C **   INDIFFERENCE ZONE(DSTAR), INITIAL SAMPLE SIZE(NAUGHT),
C **   AND THE PROBABILITY OF CORRECT SELECTION(PCS)
C **   ARE SET IN INITIALIZE BLOCK.
C **   DUDEWICZ-DALAL CRITICAL H VALUE CALCULATED BY
C **   SUBROUTINE RNKSEL



      PROGRAM AUTOH(TTY,OUTPUT,TAPE5=TTY,TAPE6=TTY,TAPE7,TAPE8,TAPE9)
      DIMENSION DATA(1000),R(1000),U(20),SIGMA(20),ITOTAL(20)
      DOUBLE PRECISION DSEED,ODSEED
C##############  INITIALIZE VARIABLES  #########################
      NAUGHT = 7
      NMAX = 0
      NDATA =100
      DELTA = 7
      PCS = .90
      KPOP = 6


C **   THE MEAN AND STANDARD DEVIATION OF THE TEST
C **   ALTERNATIVES ARE SET BELOW

      U(1) = 53
      U(2) = 60
      U(3) = 53
      U(4) = 53
      U(5) = 53.
      U(6) = 53.
      U(7) = 53.
      U(8) = 53
      U(9) = 53.
      SIGMA(1) = 4.5
      SIGMA(2) = 4.0
      SIGMA(3) = 5.5
      SIGMA(4) = 2.8
      SIGMA(5) = 6.8
      SIGMA(6) = 10.3
      SIGMA(7) = 3.4
      SIGMA(8) = 9.5
      SIGMA(9) = 8.6
      DSEED = 9841337.D0
      ODSEED = DSEED
      KPASS = 1
C######################################################
C#####  CALL SUBROUTINE TO CALCULATE  ################
```

```
C#####  DUDEWICZ AND DALAL H VALUE (DNDH)  ###########
C
C
      CALL RNKSEL(NAUGHT,KPOP,PCS,DNDH)
      WRITE(9,*)'DNDH =',DNDH
C
C#####################################################
C#####   INITIALIZE VARIABLES USED  ##################
C#####  DURING EACH REPICATION    ###################
C
C
    1 BEST = 0.0
      KOUNT = 1
    2 TSUM =  0.0
      DIFSQ = 0.0
      EXSUM = 0.0
C#############################################################
C######  GENERATE DATA --NORMAT VARIATE WITH MEAN = U  ########
C######  AND SD = SIGMA  ######################################
C
C
      CALL GGNML(DSEED,NDATA,R)
      DO 5 J=1,NDATA
        DATA(J) = (R(J)*SIGMA(KOUNT)) + U(KOUNT)
    5 CONTINUE
C
C
C#############################################################
C
C
C#####################################################
C#####    CALCULATE SAMPLE MEAN AND  #################
C#####    VARIANCE BASED ON INITIAL  #################
C#####    SAMPLE SIZE (NAUGHT)       #################
C
        DO 10 I=1,NAUGHT
          TSUM = TSUM + DATA(I)
   10    CONTINUE
        AVG = TSUM/NAUGHT
        DO 15 I=1,NAUGHT
          DIFSQ = DIFSQ + ((DATA(I)-AVG)**2)
  15    CONTINUE
        VAR = DIFSQ/(NAUGHT-1)
C
C#####################################################
C#####    DETERMINE IF LARGER SAMPLE SIZE (NEXTRA)  ###
C#####    IS NEEDED BASED ON DUDEWICZ, RAMBERG, AND  ###
C#####    CHEN PROCEDURE.  NEXTRA .GT. NAUGHT      ###
C
C
        IDDOBS=((VAR*(DNDH**2))/(DELTA**2))+.999999
        IPLUS = NAUGHT + 1
        IF (KPOP .GT. 2)THEN
          NEXTRA = MAXO(IPLUS,IDDOBS)
```

```
C **   IF REQUIRED SAMPLE SIZE(NEXTRA) IS GREATER
C **   THAN DATA AVAILABLE--RETURN AND GENERATE MORE

              IF(NEXTRA .GT. NDATA)THEN
                  WRITE(9,*)'MORE DATA NEEDED'
                  WRITE(9,*)'NEXTRA =',NEXTRA
                  NDATA=NEXTRA
                  GO TO 2
              END IF
C
C################################################
C#####   USING THE SAME PROCEDURE DETERMINE    ####
C#####   WEIGHTING FACTORS (WEIGHT) WHICH      ####
C#####   IS USED TO CALCULATE THE SPECIAL      ####
C#####   SAMPLE MEAN USED FOR BETWEEN          ####
C#####   POPULATION CONPARISONS (COMPARE)      ####
C
C
          DO 20 I=IPLUS,NEXTRA
            EXSUM = EXSUM + DATA(I)
  20      CONTINUE
          EXAVG = EXSUM/(NEXTRA-IPLUS+1)
      TOP = ((NEXTRA-NAUGHT)*(DELTA**2))
      BOTTOM = (DNDH**2)*VAR
      COR = FLOAT(NEXTRA)/FLOAT(NAUGHT)
      REC =1.0/COR
      TIM = TOP/BOTTOM
      BRAC = 1.0-TIM
      SUB = COR*BRAC
       SQ = 1.0 - SUB
      ALMOST = 1.0 + SQRT(SQ)
      WEIGHT = REC*ALMOST
       COMPARE = (WEIGHT*AVG)+((1.0-WEIGHT)*EXAVG)
      GO TO 22
      ELSE
C
C################################################
C#####   THIS SECTION IS USED IF COMPARING   #####
C#####   ONLY 2 ALTERNATIVES.  NO WEIGHTING  #####
C#####   FACTORS NEEDED                      #####
C
C
      NEXTRA = MAXO(NAUGHT,IDDOBS)
      IF(NEXTRA .GT. NDATA)THEN
          WRITE(9,*)'MORE DATA NEEDED'
          WRITE(9,*)'NEXTRA =',NEXTRA
              NDATA=NDATA+100
              GO TO 2
      END IF
      IF(NEXTRA .EQ. NAUGHT)THEN
          COMPARE = AVG
          GO TO 22
      END IF
      DO 21 I=IPLUS,NEXTRA
```

```
         TSUM = TSUM + DATA(I)
   21    CONTINUE
      COMPARE = TSUM/NEXTRA
C
C#####   END K = 2 SECTION  #####################
C###############################################
      END IF
C
C
C###############################################
C#####   WHICH POPULATION HAS THE     ###########
C#####   LARGEST 'COMPARE' VALUE AND  ###########
C#####   PRINT SUMMARY RESULTS        ###########
C
   22 IF(COMPARE .GT. BEST)THEN
          BEST = COMPARE
          NUMBER = KOUNT
      END IF
      IF(KOUNT .LT. KPOP)THEN
          DSEED = DSEED*3.D0
          KOUNT = KOUNT + 1
      IF(NEXTRA .GT. NMAX)THEN
          NMAX = NEXTRA
      END IF
          GO TO 2
      END IF
      ITOTAL(NUMBER)=ITOTAL(NUMBER) + 1
      IF(KPASS .LT. 100)THEN
          KPASS = KPASS + 1
          GO TO 1
      END IF
      DO 30 K = 1,KPOP
        WRITE(9,100)K,ITOTAL(K),U(K),SIGMA(K)
  100   FORMAT(//,6X,'ALTERNATIVE',I4,' HAS',I4,' BEST RESULTS',
     $/,6X,'THE TRUE MEAN IS',F8.2,' WITH A STANDARD DEVIATION OF',F8.2)
   30 CONTINUE
      WRITE(9,110)PCS,ODSEED,DELTA,NAUGHT,DNDH,NMAX
  110 FORMAT(///,6X,'FOR THIS TEST: PCS =',F4.3,'    DSEED =',
     $D16.8,'    DELTA =',F5.2,/,6X,'    NAUGHT =',I4,'    DNDH =',F5.3,
     $'    NMAX =',I5)

       STOP
       END


C **  SUBROUTINE TO AUTOMATICALLY GENERATE THE CRITICAL H VALUE
C **  USED TO DETERMINE THE REQUIRED SIMULATION RUN LENGTH


      SUBROUTINE RNKSEL(NO,K,PSTAR,H)
      COMMON/RSTOL/TOLF,HTOLF,NSIGD,AERR,RERR,ITMAX
      COMMON/RSCON/PI,TOLZ,BIGM
      TOLF = 1.E-6
```

```
      AERR = 1.E-8
      NSIGD = 5
      RERR = 1.E-8
      ITMAX = 500
      PI = 3.1415926535
      TOLZ = 1.E-20
      BIGM= 1.E20
      H = HO(NO,K,PSTAR,IFAULT)
      RETURN
      END
      FUNCTION HO(NNO,KK,PPSTAR,IFAULT)
      EXTERNAL GFUNC
      COMMON/RSTOL/TOLF,HTOLF,NSIGD,AERR,RERR,ITMAX
      COMMON/RSCON/PI,TOLZ,BIGM
      COMMON/RSTDST/NO,DF,K,KM1,CNORM,XPNT,A,B,PSTAR,HTEMP
      DIMENSION WK(30),PAR(1),H(1)
      DATA NDIM/1/
      NO = NNO
      K = KK
      PSTAR = PPSTAR
      HO = -BIGM
      IFAULT = 0
      DF = NO - 1
      KM1 = K - 1
      XPNT = (DF + 1.0)/2.0
      CNORM = GAMMA(XPNT)/( GAMMA(0.5*DF)*SQRT(PI*DF) )
      HTOLF = TOLF/2.0
      CALL MDSTI(HTOLF,DF,B,IER)
      IF (IER .NE. 0) THEN
         IFAULT = IER + 1000
         RETURN
      END IF
      A = -B
      H(1) = 0.0
      CALL ZSCNT(GFUNC,NSIGD,NDIM,ITMAX,PAR,H,FNORM,WK,IER)
      HO = H(1)
      IF (IER .NE. 0) THEN
      IFAULT = 2000 + IER
      END IF
      RETURN
      END
      SUBROUTINE GFUNC(H,G,NDIM,PAR)
      DIMENSION H(NDIM),G(NDIM),PAR(1)
      COMMON/RSTOL/TOLF,HTOLF,NSIGD,AERR,RERR,ITMAX
      COMMON/RSCON/PI,TOLZ,BIGM
      COMMON/RSTDST/NO,DF,K,KM1,CNORM,XPNT,A,B,PSTAR,HTEMP
      EXTERNAL SUMMND
      HTEMP = H(1)
      GTEMP = DCADRE(SUMMND,A,B,AERR,RERR,ERROR,IER)
      IF (IER.GE.100) THEN
      IFAULT = 3000 + IER
      WRITE (9,100) IFAULT
100   FORMAT (' ***ERROR IN DCADRE = ',I4,' ***')
      STOP
      END IF
```

```
      G(1) = GTEMP - PSTAR
      RETURN
      END
      FUNCTION SUMMND(T)
      COMMON/RSTOL/TOLF,HTOLF,NSIGD,AERR,RERR,ITMAX
      COMMON/RSCON/PI,TOLZ,BIGM
      COMMON/RSTDST/NO,DF,K,KM1,CNORM,XPNT,A,B,PSTAR,HTEMP
      F1 = CNORM*( (1.0 + T*T/DF)**(-XPNT) )
      X = T + HTEMP
      CALL MDTD( ABS(X), DF, TAILS, IER)
      IF (IER .GT. 0) THEN
         IFAULT = 4000 + IER
         WRITE (9,100) IFAULT
100   FORMAT (' ***ERROR IN MDTD = ',I4,' ***')
      END IF
      F2 = 0.5 + SIGN(0.5,X)*(1.0 - TAILS)
      SUMMND = F1*( F2**KM1 )
      RETURN
      END
```

APPENDIX D

```
C **  INTRODUCTION

C **  THE SPECTRAL METHOD DEVELOPED BY HEIDELBERGER AND WELCH
C **  IS USED TO ESTIMATE THE VARIANCE OF THE SAMPLE MEAN
C **  PROCEDURE USES THE ESTIMATE OF THE SPECTRUM
C **  AT ZERO FREQUENCY

      PROGRAM WELCH(TTY,OUTPUT,TAPE5=TTY,TAPE6=TTY,TAPE7,TAPE8,TAPE9)
      DIMENSION DATA(1000),R(500),PERIOD(500),XM(6),TEMP(6),B(6,7)
     $,ANOVA(16),VARB(21),FJ(500),IWK(300),WK(300),V(1000,6),VCV(21)
     $,NBR(6),ALFA(2),IJOB(2),IND(11),XYB(6,5),A(1000,6)
      DOUBLE PRECISION DSEED,P(500)
      COMPLEX TRANS(500)
      DSEED = 2003.D0
      TOALV =0.0
      TOTALZ = 0.0
      KOUNT = 0

C **  DO 25 REPLICATIONS OF TEST USING 100 DATA POINTS

      NREP = 25
  100 NDATA = 100
      KOUNT = KOUNT + 1
      U = 50.0
      SD = 30.
      NHDATA = NDATA/2
      NQDATA = NHDATA/2
      WRITE(9,*)'DSEED =',DSEED
      WRITE(9,*)'NQDATA =',NQDATA

C **  USE IMSL ROUTINE TO GENERATE NORMAL (0,1) VARIATES

      CALL GGNML(DSEED,NDATA,R)

C **  GENERATE TEST SAMPLE WITH MEAN OF U AND STANDARD
C **  DEVIATION OF SD

      DO 5 J=1,NDATA
        DATA(J) = (R(J)*SD + U)
    5 CONTINUE
      TSUM = 0.0
      VSUM = 0.0

C **  CALCULATE THE VARIANCE OF THE TEST SAMPLE USING THE
C **  CLASSICAL METHOD

      DO 10 I=1,NDATA
        TSUM = TSUM + DATA(I)
   10 CONTINUE
      AVG = TSUM/NDATA
      DO 20 I=1,NDATA
        VSUM = VSUM + ((DATA(I)-AVG)**2)
   20 CONTINUE
```

```
      CLASSV = VSUM/(NDATA-1)

C **   USE IMSL ROUTINE FFTRC TO COMPUTE THE COMPLEX
C **   CONJUGATE OF THE FAST FOURIER TRANSFORM

      CALL FFTRC(DATA,NDATA,TRANS,IWK,WK)
      MMM = NHDATA + 1
      DO 15 L=2,MMM
      K = L - 1

C **   COMPUTE PERIODOGRAM FROM FAST FOURIER TRANSFORM OF
C **   DATA (TRANS)

        PERIOD(K)=(CABS(TRANS(L))**2)/NDATA
   15 CONTINUE
      DO 30 KL=1,NQDATA
        KK =2*KL
        JJ = KK-1

C **   STABILIZE THE VARIANCE OF PERIODOGRAM BY AVERAGING
C **   OVER ADJACENT VALUES AND TAKING NATURAL LOG

      SMOTH=((PERIOD(JJ)+PERIOD(KK))/2.0)
      FJ(KL)=ALOG(SMOTH) + .270
   30 CONTINUE
      DO 50 I=1,NQDATA
          A(I,1) =  I
          A(I,2) =  I*I
          A(I,3) = I**3
          A(I,4) = I**4
          A(I,5) = I**5
          A(I,6) = FJ(I)
   50 CONTINUE
      M = 5
      IB = 6
      IND(1) = 0
      IND(2) = 0
      IND(3) = 0
      IND(4) = 0
      IND(5) = 0
      IJOB(1) = 0
      IJOB(2) = 1
      ALFA(1) = .05
      ALFA(2) = .05
      NRDIM = 1000

C **   IMSL ROUTINE TO DO FORWARD STEPWISE REGRESSION OF
C **   DATA.  USED TO SEE WHICH VARIABLES ARE IN MODEL

      CALL RLSEP(A,NQDATA,M,NRDIM,ALFA,IJOB,IND,ANOVA,XYB,IB,VARB,IER)
      WRITE(9,*)'RLSEP IER =',IER
```

```
      WRITE(9,*)'RSQR FOR RLSEP FIT',ANOVA(11)
      WRITE(9,*)'RLSEP RESULTS'
      DO 55 I=1,6
        WRITE(9,*)I,XYB(I,2)
   55 CONTINUE
      J = 0
      DO 2000 I=1,5
        IF (XYB(I,2) .NE. 0)THEN
            J = J + 1
            DO 2100 LL = 1,NQDATA
              V(LL,J) = LL**I
 2100       CONTINUE
        END IF
 2000 CONTINUE
      IF(J .EQ. 0)THEN
          ZERO = EXP(XYB(6,2))
      WRITE(9,*)'PERIODOGRAM IS A CONSTANT'
      WRITE(9,*)'U =',U,'SD =',SD
          WRITE(9,*)'SPECTRUM AT ZERO FREQ =',ZERO
          GO TO 200
      END IF
      J = J + 1
      DO 2200 I = 1,NQDATA
        V(I,J) = FJ(I)
 2200 CONTINUE
      NVAR = J
      NBR(1) = NVAR
      NBR(2) = NQDATA
      NBR(3) = NQDATA
      NBR(4) = 1
      NBR(5) = 1
      NBR(6) = 1

C **  IMSL ROUTINE TO PERFORM REQUIRED PRECONDITIONING
C **  OF DATA PRIOR TO USING RLMUL

      CALL BECOVM(V,NRDIM,NBR,TEMP,XM,VCV,IER)
      WRITE(9,*)'BECOVM IER =',IER
      IVAR = NVAR - 1
      ALPHA = 0.05


C **  IMSL ROUTINE FOR STANDARD MULTIPLE LINEAR REGRESSION
C **  SUPPLIES AS OUTPUTS THE VALUES OF RESIDUAL MEAN
C **  SQUARE AND THE ESTIMATED STANDARD ERROR OF
C **  THE INTERCEPT. REQUIRED TO CALCULATE THE UPPER
C **  LEFT ELEMENT OF (X'X)-1 MATRIX

      CALL RLMUL(VCV,XM,NQDATA,IVAR,ALPHA,ANOVA,B,IB,VARB,IER)
      DO 60 I = 1,NVAR
          WRITE(9,*)'RLMUL COEFF',I,' = ',B(I,1)
   60 CONTINUE
      WRITE(9,*)'BOSE =  ',B(NVAR,4),'ANOVA8 = ',ANOVA(8)
      UPLEFT = (B(NVAR,4)**2)/ANOVA(8)
      WRITE(9,*)'ANOVA11 =',ANOVA(11)
```

```
      WRITE(9,*)'U =',U,'SD =',SD
      WRITE(9,*)'UPLEFT = ',UPLEFT
      ADJUST = (.645*UPLEFT)/2.0
      CONE = EXP(-ADJUST)
      ZERO = CONE*(EXP(B(NVAR,1)))
      WRITE(9,*)UPLEFT,CONE,ZERO
  200 WRITE(9,*)'CLASSV DURING KOUNT',CLASSV,KOUNT

C **  AVERAGE THE MEASURED VARIANCE BY CLASSICAL AND SPECTRAL
C **  METHOD OVER THE 25 RUNS AND PRINT RESULTS

      TOTALV = TOTALV + CLASSV
      TOTALZ = TOTALZ + ZERO
      IF (KOUNT .LT. NREP)GO TO 100
      AVGSD = TOTALV/NREP
      AVGZ = TOTALZ/NREP
      WRITE(9,*)'AVGSD=',AVGSD,'AVGZ=',AVGZ
      STOP
      END
```

APPENDIX E

```
C **  THIS PROGRAM IS A SLAM SIMULATION USED TO MODEL A SERIAL
C **  QUEUE REPAIR FACILITY.  THE NUMBER OF REPAIR STATIONS
C **  IS SET BY THE VARIABLE NSTA.  THE ARRIVAL RATE AND
C **  SERVICE RATE ARE SET BY ARIVAL AND SERVIC RESPECTIVELY

C **  THE STEADY STATE PERFORMANCE MEASURE FOR COMPARING
C **  THE ALTERNATIVE CONFIGURATIONS IS CALCULATED BY THE
C **  INTEGRATED MULTIPLE RANKING PROCEDURE DEVELOPED BY
C **  J WILSON AND T DICKINSON AT THE UNIVERSITY OF TEXAS


C **  THE CODE WRITTEN STARTING AT SUBROUTINE NQUE MY BE LIFTED
C **  AND USED AS A MULTIPLE RANKING PROCEDURE FOR ANY DISCRETE
C **  EVENT SIMULATION WRITTEN USING SLAM

C **  IT IS POSSIBLE WITH MINOR MODIFICATION TO USE THIS PROCEDURE
C **  ON A SIMULAION UTILIZING ANOTHER LANGUAGE

      PROGRAM MAIN(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,TAPE7,TAPE8)
      COMMON QSET(5000)
      COMMON/SCOM1/ ATRIB(100),DD(100),DDL(100),DTNOW,II,MFA,MSTOP,NCLNR
     1,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,SS(100),SSL(100),TNEXT,TNOW,XX(100)
     2,ARIVAL,SERVIC,NSTRM,KOUNT,N,NSIZE,ALPHA,NSTA,KKK,LENROW,NREP,IEXP
C
      NNSET = 5000
      NCRDR = 5
      NPRNT = 6
      NTAPE = 7
      NSTRM = 3
      KOUNT = 0
      ARIVAL = 2.0
      SERVIC = 1.0
      LENROW = 100
      NREP = 16
      NSIZE = 1
      ALPHA = .1
      NSTA = 1
      CALL SLAM
      STOP
      END


C **  SLAM SUBROUTINE TO SET INITAL CONDITIONS FOR THE SIMULATION

      SUBROUTINE INTLC
      COMMON/SCOM1/ ATRIB(100),DD(100),DDL(100),DTNOW,II,MFA,MSTOP,NCLNR
     1,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,SS(100),SSL(100),TNEXT,TNOW,XX(100)
     2,ARIVAL,SERVIC,NSTRM,KOUNT,N,NSIZE,ALPHA,NSTA,KKK,LENROW,NREP,IEXP
```

```
C
C**** SET UP INITIAL CONDITIONS FOR MODEL
C
C****  TELLER STATUS
C
      DO 5 I=1,NSTA
        XX(I) = 0.0
    5 CONTINUE
C
C
C
C**** INITIALIZE KKK FOR EACH RUN
      KKK = 0
      N = 1
      IF(MOD(NNRUN,16) .NE. 0)THEN
           IMUL = NNRUN/16
           IEXP = IMUL + 1
           KOUNT = NNRUN - (IMUL*16)
      ELSE
           KOUNT = 16
           IEXP = NNRUN/16
      END IF
      IF (KOUNT .EQ.1)THEN
           WRITE(6,*)' EXPERIMENT NUMBER =',IEXP
      END IF
C
C**** SET UP ATTRIBUTES OF FIRST ARRIVING XACT
C
      TNA = EXPON(ARIVAL,NSTRM)
C
C     ARRIVAL TIME
      ATRIB(1) = TNOW + TNA
C
C     SERVICE TIME
      ATRIB(2) = EXPON(SERVIC,NSTRM)
C
C     SERIAL NUMBER
      ATRIB(3) = N
C
C****  SHOW NEXT SERVICE FACILITY TO VISIT
C
C
      ATRIB(4) = 1
C
C
C
C**** POST ENTRY ON EVENT CALENDAR WITH EVENT CODE = 1, DELAY = TNA
C
      CALL SCHDL(1,TNA,ATRIB)
      RETURN
      END
      SUBROUTINE EVENT(ICODE)
C
C**** INVOKE APPROPRIATE EVENT PROCESSING ROUTINE
C
```

```
      GO TO (10,20,30),ICODE
C
C     ARRIVAL EVENT
   10 CALL ARVL
      RETURN
C
C     END OF SERVICE EVENT
   20 CALL ENDSV
      RETURN
C
C
C       DATA EVENT
   30 CALL NQUE
      RETURN
      END
C


C **  SLAM SUBROUTINE TO SCHEDULE ARRIVALS TO NEXT WORK STATION
C **  INCLUDES SCHEDULING NEW ARRIVALS TO THE SYSTEM


C
      SUBROUTINE ARVL
      COMMON/SCOM1/ ATRIB(100),DD(100),DDL(100),DTNOW,II,MFA,MSTOP,NCLNR
     1,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,SS(100),SSL(100),TNEXT,TNOW,XX(100)
     2,ARIVAL,SERVIC,NSTRM,KOUNT,N,NSIZE,ALPHA,NSTA,KKK,LENROW,NREP,IEXP
      DIMENSION BUFFR(7)
C
C**** GENERATE THE NEXT ARRIVAL BEFORE PROCESSING THE CURRENT XACT
C
      IF (ATRIB(4) .GT. 1) GO TO 5
      TNA = EXPON(ARIVAL,NSTRM)
      BUFFR(1) = TNOW + TNA
      BUFFR(2) = EXPON(SERVIC,NSTRM)
      N = N + 1
      BUFFR(3) = N
      BUFFR(4) = 1
      CALL SCHDL(1,TNA,BUFFR)
C
C**** DETERMINE DISPOSITION OF CURRENT ARRIVAL
C
    5 MMM = ATRIB(4)
C
C     CHECK SERVER STATUS
      IF (XX(MMM).EQ.1.0) GO TO 10
C
C       IF IDLE, GO INTO SERVICE
      XX(MMM) = 1.0
      SVCTIM = ATRIB(2)
C
C     POST ENTRY ON EVENT CALENDAR WITH EVENT CODE = 2, DELAY = SVCTIM
      CALL SCHDL(2,SVCTIM,ATRIB)
      RETURN
```

```
C
C       IF TELLER IS BUSY, GO INTO QUEUE
  10      CALL FILEM(MMM,ATRIB)
        RETURN
        END



C **    SUBROUTINE TO PROCESS CUSTOMERS AS THEY FINISH SERVICE AT
C **    WORK STATION.  IF STATION IS LAST, CALL SUBROUTINE(NQUE) TO
C **    COLLECT NECESSARY DATA


        SUBROUTINE ENDSV
        COMMON/SCOM1/ ATRIB(100),DD(100),DDL(100),DTNOW,II,MFA,MSTOP,NCLNR
       1,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,SS(100),SSL(100),TNEXT,TNOW,XX(100)
       2,ARIVAL,SERVIC,NSTRM,KOUNT,N,NSIZE,ALPHA,NSTA,KKK,LENROW,NREP,IEXP
        DIMENSION BUFFR(7)
C
C**** PROCESS DEPARTING CUSTOMER
C
        JJ = ATRIB(4)
        IF (ATRIB(4) .GE. NSTA) GO TO 20
        ATRIB(4) = ATRIB(4) + 1
        ATRIB(2) = EXPON(SERVIC,NSTRM)
        CALL SCHDL(1,0.,ATRIB)
        GO TO 30
   20 ATRIB(5) = TNOW - ATRIB(1)
        CALL SCHDL(3,0.,ATRIB)
C
C**** DETERMINE DISPOSITION OF SERVER
C
C       CHECK THE QUEUE
   30 IF (NNQ(JJ) .EQ. 0) GO TO 10
C
C       IF NONEMPTY, REMOVE FIRST XACT FROM FILE JJ & PUT INTO SERVICE
        CALL RMOVE(1,JJ,BUFFR)
        SVCTIM = BUFFR(2)
C
C       POST ENTRY ON EVENT CALENDAR WITH EVENT CODE = 2, DELAY = SVCTIM
        CALL SCHDL(2,SVCTIM,BUFFR)
        RETURN
C
C       IF QUEUE IS EMPTY, CHANGE TELLER STATUS
  10    XX(JJ) = 0.0
        RETURN
        END
C
C
C


C **    SUBROUTINE TO COLLECT DATA AND CALL APPROPRIATE SUBROUTINE
```

```
C **  WHEN INITIAL SIMULATION RUNS ARE COMPLETE--CALLS SUBROUTINES
C **  TO INDUCE NORMALITY(WILK), DETECT AND TRUNCATE INITIAL BIAS
C **  (IBSUB), AND CALCULATE REQUIRED SIMULATION RUN LENGTH(MRPSUB)
C **  BASED ON THIS RUN LENGTH THE ROUTINE CALCULATES
C **  STATISTIC (COMPARE) USED TO RANK ALTERNATIVES


C
C
C
      SUBROUTINE NQUE
      COMMON/SCOM1/ ATRIB(100),DD(100),DDL(100),DTNOW,II,MFA,MSTOP,NCLNR
     1,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,SS(100),SSL(100),TNEXT,TNOW,XX(100)
     2,ARIVAL,SERVIC,NSTRM,KOUNT,N,NSIZE,ALPHA,NSTA,KKK,LENROW,NREP,IEXP
      DIMENSION BUFFR(7),ADD(30000),DATA(16,100),BATCH(10000)
     $,DNORMAL(5000),ARRAYC(100)


      KKK = KKK + 1
      IF ( IFAULT .EQ. 50) GO TO 1990


      IF(KKK .GT. LENROW)GO TO 7
      IBDATA = LENROW
      DATA(KOUNT,KKK) = -ATRIB(5)
      IF(KOUNT.EQ.NREP.AND.KKK.EQ.LENROW)THEN
          LLL = 0
          CALL SWSUB(DATA,NREP,NSIZE,IFAULT)



      IF (NSIZE .GT. 3 .AND. IFAULT .NE. 10)THEN
          WRITE(6,*)' PASSED SW TEST BUT NSIZE TOO BIG'
          NSIZE = 3
      END IF
      IF (IFAULT .EQ. 10)THEN
         WRITE(6,*)' PROBLEMS NSIZE TOO BIG'
         WRITE(6,*)'CONTINUE WITH NSIZE = 3'
         NSIZE = 3
      END IF
      DO 1970 I=1,LENROW
        DNORMAL(I) = DATA(KOUNT,I)
 1970 CONTINUE
      CALL IBSUB(DNORMAL,NSIZE,LENROW,ALPHA,ITRUNC,IFAULT
     $,BATCH,NUMBAT)
 1980 JUNK = ITRUNC*NSIZE
      IF (IFAULT .EQ.50) THEN
          WRITE(6,*)'TOO MUCH TRUNCATION'
          RETURN
 1990     LLL = LLL + 1
          IBEX = LENROW + LLL
          DNORMAL(IBEX) = -ATRIB(5)
          IBDATA = (2.5*JUNK)
          IF (IBEX .LT. IBDATA)RETURN
```

```
          CALL IBSUB (DNORMAL,NSIZE,IBDATA,ALPHA,ITRUNC
     $    ,IFAULT,BATCH,NUMBAT)
          GO TO 1980
      END IF
          WRITE(6,2000)ALPHA,IBDATA,JUNK,NSIZE,NUMBAT
 2000     FORMAT(//,7X,'BASED ON TYPE I ERROR OF',F6.3,
     $       ' THE ORIGINAL',I6,' DATA POINTS MUST ',/,7X,
     $       'HAVE',I4,' DATA POINTS TRUNCATED TO ELIMINATE IB'
     $       ,/,7X,'USING A BATCH SIZE OF',I4,' THERE ARE',/,
     $       ,I6,' USEABLE BATCHED DATA POINTS FOR FURTHER TESTING')



          CALL MRPSUB (NUMBAT,BATCH,NEXTRA,IFAULT)


          IF (IFAULT .EQ. 100) THEN
              WRITE(6,*)' '
          END IF
          IF (NEXTRA .EQ. NUMBAT) THEN
              TSUM = 0.0
              DO 2010 I=1,NUMBAT
                TSUM = TSUM + BATCH(I)
 2010         CONTINUE
              COMPARE = TSUM/NUMBAT
              WRITE(6,*)' FINAL COMPARE VALUE =',COMPARE
              WRITE(6,1234)
 1234         FORMAT(////)
              GO TO 30
          END IF
      IF(KKK .EQ. IBDATA)RETURN
    7 ADD(KKK) = -ATRIB(5)
      JJJ = (NEXTRA - NUMBAT)*NSIZE + IBDATA
      IF (JJJ .GT. NSAMPLE)NSAMPLE = JJJ
      IF(KKK .LT. JJJ)RETURN
C
C
      IBEGIN = IBDATA + 1
      IFIN = IBDATA + NSIZE
      NDIF = NEXTRA - NUMBAT
      III = NUMBAT
C
      DO 10 I=1,NDIF
        TOTAL = 0.0
        DO 15 J=IBEGIN,IFIN
          TOTAL = TOTAL + ADD(J)
   15   CONTINUE
C
        III = III + 1
        BATCH(III) = TOTAL/NSIZE
        IBEGIN = IBEGIN + NSIZE
        IFIN = IFIN + NSIZE
```

```
   10 CONTINUE
              FINALT = 0.0
              DO 20 I = 1,NEXTRA
                FINALT = FINALT + BATCH(I)
   20         CONTINUE
              COMPARE = FINALT/NEXTRA
              ARRAYC(IEXP) = COMPARE
              WRITE(6,*)'USING A TOTAL OF',NEXTRA,' BATCHED POINTS'
              WRITE(6,*)'A FINAL COMPARE VALUE =',COMPARE
               WRITE(6,*)'FINAL NUMBER OF DATA POINTS=',JJJ
              WRITE(6,1234)
              IF(IEXP .EQ. 50 .AND. KOUNT .EQ. 16)THEN
                  DO 25 I=1,IEXP
                    WRITE(6,*)I,ARRAYC(I)
   25             CONTINUE
                  WRITE(6,*)' BIGGEST SAMPLE SIZE =',NSAMPLE
              END IF
              MSTOP = -1
              RETURN
      END IF
   30 IF(KKK .EQ. LENROW)MSTOP = -1
      RETURN
      END
C
C **  SUBROUTINE TO DETERMINE REQUIRED BATCH SIZE TO INDUCE
C **  NORMALITY IN DATA


C
      SUBROUTINE SWSUB(DATA,NREP,NSIZE,IFAULT)
C
C
C     SHAPIRO-WILK TEST FOR NORMALITY
C
      COMMON /ABC/ J,K,SSQ,INDCOL(50),X(200),W
      DATA INDCOL/0,0,0,1,3,5,8,11,15,19,24,29,35,41,48,55,63,71,80,89,9
     19,109,120,131,143,155,168,181,195,209,224,239,255,271,288,305,323,
     2341,360,379,399,419,440,461,483,505,528,551,575,599/
      DIMENSION DATA(16,100)
C
C    INITIALIZE VARIALBLES
C
      IFAULT = 0
      IPR = 0
      NSIZE = 1
      N = NREP
C
C
 1111 DO 10 I = 1,NREP
        SUM = 0.0
        DO 20 J = 1,NSIZE
          SUM = SUM + DATA(I,J)
   20   CONTINUE
        AVG = SUM/NSIZE
```

```
          X(I) = AVG
   10 CONTINUE
C
C
C
C
C     SORT DATA POINTS IN ASCENDING ORDER
C
      CALL QSORT (X,N)
C
C     CALCULATIONS
C
      IF (N.LE.50) J=INDCOL(N)
      K=N/2
      SUM=0.
      SSQ=0.
      DO 60 I=1,N
         SUM=SUM+X(I)
   60 SSQ=SSQ+X(I)*X(I)
      SSQ=SSQ-SUM*SUM/N
C
C     PRINT ORDERED DATA IF NEEDED
C
      IF (IPR.NE.0) GO TO 70
C
C     GET W STATISTIC
C
   70 CALL TEST(NREP)
C
C
C      PRINT CRITICAL VALUES IF NEEDED
C
      CALL CRITVAL(N,CRIT)
      IF(W .LT. CRIT)THEN
          IF (NSIZE .EQ. 10)THEN
              WRITE(6,*)'BATCH SIZE OF 10 FAILS SHAPIRO-WILK TEST '
              WRITE(6,*)'TO DETERMINE ADEQUATE VARIANCE MEASURE'
              WRITE(6,*)'A LARGER INITIAL SAMPLE IS NEEDED'
              IFAULT = 10
              RETURN
          END IF
          NSIZE = NSIZE + 1
          GO TO 1111
      END IF
      WRITE(6,*)'PASSED THE SW TEST WITH NSIZE =',NSIZE
      WRITE(6,*)'CALCULATED W =',W,' CRITICAL VALUE =',CRIT
C
  190 FORMAT (//)
C
C
      RETURN
      END
      SUBROUTINE TEST(NREP)
C
C     THIS SUBROUTINE CALCULATES THE SHAPIRO-WILK W STATISTIC
```

```
C
      COMMON /ABC/ J,K,SSQ,INDCOL(50),X(200),W
      COMMON /COEF/ A(625)
C
      B=0
      N = NREP
      DO 10 I=1,K
   10 B=B+A(J+I)*(X(N-I+1)-X(I))
      W=B*B/SSQ
      RETURN
C
C
      END
      SUBROUTINE CRITVAL (N,CRIT)
C
C     PRINT THE CRITICAL VALUES FOR THE W TEST
C
      COMMON /CRIT/ T(50,1)
C
      CRIT = T(N,1)
      RETURN
C
C
      END
      BLOCK DATA W
      COMMON /COEF/ A(200),C(200),D(200),F(25)
      COMMON /CRIT/ T(50,1)
      DATA A/.7071,.6872,.1677,.6646,.2413,.6431,.2806,.0875,.6233,.3031
     1,.1401,.6052,.3164,.1743,.0561,.5888,.3244,.1976,.0947,.5739,.3291
     2,.2141,.1224,.0399,.5601,.3315,.2260,.1429,.0695,.5475,.3325,.2347
     3,.1586,.0922,.0303,.5359,.3325,.2412,.1707,.1099,.0539,.5251,.3318
     4,.2460,.1802,.1240,.0727,.0240,.5150,.3306,.2495,.1878,.1353,.0880
     5,.0433,.5056,.3290,.2521,.1939,.1447,.1005,.0593,.0196,.4968,.3273
     6,.2540,.1988,.1524,.1109,.0725,.0359,.4886,.3253,.2553,.2027,.1587
     7,.1197,.0837,.0496,.0163,.4808,.3232,.2561,.2059,.1641,.1271,.0932
     8,.0612,.0303,.4734,.3211,.2565,.2085,.1686,.1334,.1013,.0711,.0422
     9,.0140,.4643,.3185,.2578,.2119,.1736,.1399,.1092,.0804,.0530,.0263
     *,.4590,.3156,.2571,.2131,.1764,.1443,.1150,.0878,.0618,.0368,.0122
     *,.4542,.3126,.2563,.2139,.1787,.1480,.1201,.0941,.0696,.0459,.0228
     *,.4493,.3098,.2554,.2145,.1807,.1512,.1245,.0997,.0764,.0539,.0321
     *,.0107,.4450,.3069,.2543,.2148,.1822,.1539,.1283,.1046,.0823,.0610
     *,.0403,.0200,.4407,.3043,.2533,.2151,.1836,.1563,.1316,.1089,.0876
     *,.0672,.0476,.0284,.0094,.4366,.3018,.2522,.2152,.1848,.1584,.1346
     *,.1128,.0923,.0728,.0540,.0358,.0178,.4328,.2992,.2510,.2151,.1857
     *,.1601,.1372,.1162,.0965,.0778,.0598,.0424,.0253,.0084,.4291,.2968
     *,.2499,.2150/
      DATA C/.1864,.1616,.1395,.1192,.1002,.0822,.065,.0483,.032,.0159,.
     14254,.2944,.2487,.2148,.1870,.1630,.1415,.1219,.1036,.0862,.0697,.
     20537,.0381,.0227,.0076,.4220,.2921,.2475,.2145,.1874,.1641,.1433,.
     31243,.1066,.0899,.0739,.0585,.0435,.0289,.0144,.4188,.2898,.2463,.
     42141,.1878,.1651,.1449,.1265,.1093,.0931,.0777,.0629,.0485,.0344,.
     50206,.0068,.4156,.2876,.2451,.2137,.1880,.1660,.1463,.1284,.1118,.
     60961,.0812,.0669,.0530,.0395,.0262,.0131,.4127,.2854,.2439,.2132,.
     71882,.1667,.1475,.1301,.1140,.0988,.0844,.0706,.0572,.0441,.0314,.
     80187,.0062,.4096,.2834,.2427,.2127,.1883,.1673,.1487,.1317,.1160,.
```

```
     91013,.0873,.0739,.0610,.0484,.0361,.0239,.0119,.4068,.2813,.2415,.
     *2121,.1883,.1678,.1496,.1331,.1179,.1036,.0900,.0770,.0645,.0523,.
     *0404,.0287,.0172,.0057,.4040,.2794,.2403,.2116,.1883,.1683,.1505,.
     *1344,.1196,.1056,.0924,.0798,.0677,.0559,.0444,.0331,.0220,.0110,.
     *4015,.2774,.2391,.2110,.1881,.1686,.1513,.1356,.1211,.1075,.0947,.
     *0824,.0706,.0592,.0481,.0372,.0264,.0158,.0053,.3989,.2755,.2380,.
     *2104,.1880,.1689,.1520,.1366,.1225,.1092,.0967,.0848,.0733,.0622,.
     *0515,.0409,.0305,.0203,.0101,.3964,.2737,.2368,.2098,.1878,.1691,.
     *1526,.1376,.1237,.1108,.0986,.0870,.0759,.0651,.0546,.0444,.0343,.
     *0244,.0146,.0049/
      DATA D/.394,.2719,.2357,.2091,.1876,.1693,.1531,.1384,.1249,.1123,
     1.1004,.0891,.0782,.0677,.0575,.0476,.0379,.0283,.0188,.0094,.3917,
     2.2701,.2345,.2085,.1874,.1694,.1535,.1392,.1259,.1136,.1020,.0909,
     3.0804,.0701,.0602,.0506,.0411,.0318,.0227,.0136,.0045,.3894,.2684,
     4.2334,.2078,.1871,.1695,.1539,.1398,.1269,.1149,.1035,.0927,.0824,
     5.0724,.0628,.0534,.0442,.0352,.0263,.0175,.0087,.3872,.2667,.2323,
     6.2072,.1868,.1695,.1542,.1405,.1278,.1160,.1049,.0943,.0842,.0745,
     7.0651,.0560,.0471,.0383,.0296,.0211,.0126,.0042,.3850,.2651,.2313,
     8.2065,.1865,.1695,.1545,.1410,.1286,.1170,.1062,.0959,.0860,.0765,
     9.0673,.0584,.0497,.0412,.0328,.0245,.0163,.0081,.3830,.2635,.2302,
     *.2058,.1862,.1695,.1548,.1415,.1293,.1180,.1073,.0972,.0876,.0783,
     *.0694,.0607,.0522,.0439,.0357,.0277,.0197,.0118,.0039,.3808,.2620,
     *.2291,.2052,.1859,.1695,.1550,.14[illegible]0,.1300,.1189,.1085,.0986,.0892,
     *.0801,.0713,.0628,.0546,.0465,.0385,.0307,.0229,.0153,.0076,.3789,
     *.2604,.2281,.2045,.1855,.1693,.1551,.1423,.1306,.1197,.1095,.0998,
     *.0906,.0817,.0731,.0648,.0568,.0489,.0411,.0335,.0259,.0185,.0111,
     *.0037,.3770,.2589,.2271,.2038,.1851,.1692,.1553,.1427,.1312,.1205,
     *.1105,.1010,.0919,.0832,.0748,.0667,.0588,.0511,.0436,.0361,.0288,
     *.0215,.0143,.0071/
      DATA F/.3751,.2574,.226,.2032,.1847,.1691,.1554,.143,.1317,.1212,.
     11113,.1020,.0932,.0846,.0764,.0685,.0608,.0532,.0459,.0386,.0314,.
     20244,.0174,.0104,.0035/
       DATA T/0.,0.,.789,.792,.806,.826,.838,.851,.859,.869,.8
     *76,.883,.889,.895,.901,.906,.910,.914,.917,.920,.923,.926,.928,.93
     *0,.931,.933,.935,.936,.937,.939,.940,.941,.942,.943,.944,.945,.946
     *,.947,.948,.949,.950,.951,.951,.952,.953,.953,.954,.954,.955,.955/
C
      END
      SUBROUTINE QSORT (X,N)
C
C     QUICKSORT ALGORITHM.
C
      DIMENSION X(1), STACK(13,2)
      INTEGER STACK,FIRST
      REAL MEDIAN,MED
      DATA MAXSTK/13/,M/10/
      ITOP=0
      FIRST=1
      NN=N
   10 CONTINUE
      IF (NN.GT.M) GO TO 20
      CALL SHLSRT (X(FIRST),NN)
      IF (ITOP.LE.0) GO TO 130
      FIRST=STACK(ITOP,1)
      NN=STACK(ITOP,2)
```

```
      ITOP=ITOP-1
      GO TO 10
   20 CONTINUE
      LAST=FIRST+NN-1
      N1=0
      N2=0
      MEDIAN=MED(X(FIRST),NN)
      I1=FIRST
      I2=LAST+1
   30 CONTINUE
      I=I2-1
   40 CONTINUE
      IF (I.LE.I1) GO TO 80
      IF (X(I).LT.MEDIAN) GO TO 50
      I=I-1
      N2=N2+1
      GO TO 40
   50 I2=I
      X(I1)=X(I2)
      N1=N1+1
      I=I1+1
   60 CONTINUE
      IF (I.GE.I2) GO TO 90
      IF (X(I).GT.MEDIAN) GO TO 70
      I=I+1
      N1=N1+1
      GO TO 60
   70 I1=I
      X(I2)=X(I1)
      N2=N2+1
      GO TO 30
   80 X(I1)=MEDIAN
      GO TO 100
   90 X(I2)=MEDIAN
  100 CONTINUE
      ITOP=ITOP+1
      IF (ITOP.GT.MAXSTK) GO TO 120
      IF (N1.GT.N2) GO TO 110
      STACK(ITOP,1)=LAST-N2+1
      STACK(ITOP,2)=N2
      NN=N1
      GO TO 10
  110 STACK(ITOP,1)=FIRST
      STACK(ITOP,2)=N1
      FIRST=LAST-N2+1
      NN=N2
      GO TO 10
  120 CALL REMARK (24LSTACK OVERFLOW IN QSORT )
      STOP
  130 RETURN
C
      END
      SUBROUTINE SHLSRT (X,N)
C
C     SHELL SORT.
```

```
C
      DIMENSION X(1)
      M=N
   10 CONTINUE
      M=M/2
      IF (M.EQ.0) GO TO 50
      K=N-M
      J=1
   20 CONTINUE
      I=J
   30 CONTINUE
      L=M+I
      IF (X(I).LE.X(L)) GO TO 40
      XK=X(I)
      X(I)=X(L)
      X(L)=XK
      I=I-M
      IF (I.GE.1) GO TO 30
   40 CONTINUE
      J=J+1
      IF (J-K) 20,20,10
   50 CONTINUE
      RETURN
C
      END
      FUNCTION MED (X,N)
C
C     FUNCTION TO GET A MEDIAN ESTIMATE OF AN ARRAY.
C
      REAL MED
      DIMENSION X(1)
      MID=N/2
      XF=X(1)
      XM=X(MID)
      XL=X(N)
      IF (XF.GT.XM) GO TO 10
      IF (XM.LT.XL) GO TO 30
      IF (XF.LT.XL) GO TO 40
      GO TO 20
   10 IF (XM.GT.XL) GO TO 30
      IF (XF.GT.XL) GO TO 40
   20 K=1
      GO TO 50
   30 K=MID
      GO TO 50
   40 K=N
   50 MED=X(K)
      X(K)=X(1)
      RETURN
C
      END
C
C
```

```
**************************  END OF SUBROUTINE WILK  *********************


C
C
C
C
C   ****THIS IS THE SUBROUTINE THAT CALCULATES TRUNCATION POINT
C   ****NEEDED TO ELIMINATE ANY DETECTED INITIAL BIAS
C
      SUBROUTINE IBSUB (DNORMAL,NSIZE,LENROW,ALPHA,ITRUNC,IFAULT
     $,BATCH,NUMBAT)
      DIMENSION BATCH(10000),DNORMAL(5000)
C   ******PERFORM SCHRUBEN INITIALIZATION BIAS TEST
C
C
C   *****INITIALIZE VARIABLES
C
      LENTH = LENROW
      OBS = LENTH
      JUNK = 0
      IBEGIN = 1
      MPOINT = 0
      IEND = NSIZE
      ITRUNC = 0
      KOUNT = 0
C
C
C
C
   75 NUMBAT = OBS/NSIZE
      NSTART = IBEGIN
      POINTS = FLOAT(NUMBAT)
      KOUNT = KOUNT + 1
      NFIN = IEND
      DO 150 K=1,NUMBAT
      BATSUM = 0.0
      DO 100 I=NSTART,NFIN
      BATSUM = BATSUM + DNORMAL(I)
  100 CONTINUE
C
C   *****COMPUTE MEAN OF EACH BATCH
C
      BATCH(K) = BATSUM/NSIZE
      NSTART = NSTART + NSIZE
      NFIN = NFIN + NSIZE
  150 CONTINUE
C
C
C   *****COMPUTE SUM OF SAFE BATCH MEANS
C
      TOT = 0.0
      NHALF = NUMBAT/2
```

```
      MID = NHALF + 1
      DO 200  I=MID,NUMBAT
        TOT = TOT + BATCH(I)
  200 CONTINUE
C
C
C
C  ****CALL TO SUBROUTINE TO CALCULATE VARIANCE BY WELCH PROCEDURE
C
      CALL WELCH(BATCH,MID,NUMBAT,ZERO)
C
      GAMMA = ZERO
C
C
C
C  *****INITIALIZE VARIABLES USED DURING
C  *****EACH PASS THROUGH THE DATA
C
      AMAX = 0.0
      CUSUM = 0.0
      PMEAN = 0.0
      AMIN = 0.0
      PSUM = 0.0
      M = 0
      NEGTIV = 0
      POSTIV = 0
      TOTAL = 0.0
C
C  *****COMPUTE MEAN OF ALL DATA
C
      DO 325  I = 1,NUMBAT
        TOTAL = TOTAL + BATCH(I)
  325 CONTINUE
      AMEAN = TOTAL/NUMBAT
C
C
C  *****TEST FOR INITIALIZATION BIAS
C  *****FIND MOST POSITIVE AND NEGATIVE
C  *****VALUES OF NORMALIZED CUSUM
C
      SQROOT = SQRT(POINTS)
      DO 500  I=1,NUMBAT
        M = M+1
        PSUM = PSUM+BATCH(I)
        PMEAN = PSUM/M
        CUSUM = AMEAN-PMEAN
C
C  *****BLOCK TO CHECK FOR NEGATIVE VALUES OF CUSUM
C  *****AND SAVE MOST NEGATIVE VALUE
C
      IF(CUSUM .LT. 0.0)THEN
          NEGTIV = 1
          SNEG = (M*CUSUM)/SQROOT
          IF(SNEG.LT.AMIN)THEN
              AMIN=SNEG
```

```
              NEGLOC = M
          END IF
          GO TO 500
      END IF
C
C
        STAR = (M*CUSUM)/SQROOT
        IF(STAR .GT. AMAX)THEN
            POSTIV = 1
            AMAX = STAR
            MPOINT = M
        END IF
  500 CONTINUE
C
C   *****BLOCK TO CHECK IF ONLY POSITIVE
C   *****INITIAL BIAS INDICATED
C
      IF((NEGTIV .GT. 0) .AND.
     +(POSTIV .LT. 1.0))THEN
          AMAX = AMIN
          MPOINT =NEGLOC
          GO TO 501
      END IF
C
C   *****BLOCK TO CHECK IF OSCILLATION OF
C   *****NEGATIVE AND POSITIVE BIAS INDICATED.
C
C   *****STANDARDIZE TO UNIT INTERVAL.
C
C   *****CHECK USING SAME SCHRUBEN TEST
C   *****EXCEPT USING ALPHA/2.
C
      IF((NEGTIV .GT. 0).AND.(POSTIV.GT.0))THEN
          TN = FLOAT(NEGLOC)/POINTS
          TP = FLOAT(MPOINT)/POINTS
          XP=(AMAX**2)/(3*GAMMA*TP*(1-TP))
          XN=(AMIN**2)/(3*GAMMA*TN*(1-TN))
          DFN =3.
          DFD=POINTS/2
          X = XP
          CALL MDFDRE(X,DFN,DFD,P,IER)
          PROPOS = 1. - P
          X =XN
          CALL MDFDRE(X,DFN,DFD,P,IER)
          PRONEG = 1.0 - P
          HALPHA = ALPHA/2.
          IF((PROPOS .AND. PRONEG)
     +     .LT. HALPHA)THEN
              MPOINT=MAX0(MPOINT,NEGLOC)
              GO TO 913
          END IF
          IF((PROPOS .AND. PRONEG)
     +     .GE. HALPHA)THEN
              GO TO 974
          END IF
```

```
              NEGLOC = M
          END IF
          GO TO 500
      END IF
C
C
        STAR = (M*CUSUM)/SQROOT
        IF(STAR .GT. AMAX)THEN
            POSTIV = 1
            AMAX = STAR
            MPOINT = M
        END IF
  500 CONTINUE
C
C   *****BLOCK TO CHECK IF ONLY POSITIVE
C   *****INITIAL BIAS INDICATED
C
      IF((NEGTIV .GT. 0) .AND.
     +(POSTIV .LT. 1.0))THEN
          AMAX = AMIN
          MPOINT =NEGLOC
          GO TO 501
      END IF
C
C   *****BLOCK TO CHECK IF OSCILLATION OF
C   *****NEGATIVE AND POSITIVE BIAS INDICATED.
C
C   *****STANDARDIZE TO UNIT INTERVAL.
C
C   *****CHECK USING SAME SCHRUBEN TEST
C   *****EXCEPT USING ALPHA/2.
C
      IF((NEGTIV .GT. 0).AND.(POSTIV.GT.0))THEN
          TN = FLOAT(NEGLOC)/POINTS
          TP = FLOAT(MPOINT)/POINTS
          XP=(AMAX**2)/(3*GAMMA*TP*(1-TP))
          XN=(AMIN**2)/(3*GAMMA*TN*(1-TN))
          DFN =3.
          DFD=POINTS/2
          X = XP
          CALL MDFDRE(X,DFN,DFD,P,IER)
          PROPOS = 1. - P
          X =XN
          CALL MDFDRE(X,DFN,DFD,P,IER)
          PRONEG = 1.0 - P
          HALPHA = ALPHA/2.
          IF((PROPOS .AND.PRONEG)
     +     .LT. HALPHA)THEN
              MPOINT=MAXO(MPOINT,NEGLOC)
              GO TO 913
          END IF
          IF((PROPOS .AND. PRONEG)
     +     .GE. HALPHA)THEN
              GO TO 974
          END IF
```

```
      IF(PROPOS .LT. HALPHA)THEN
              GO TO 913
          ELSE
              MPOINT = NEGLOC
              GO TO 913
          END IF
      END IF
C
C  *****BLOCK TO CALCULATE VIA SCHRUBEN
C  *****BROWNIAN BRIDGE TEST INDICATION
C  *****OF INITIAL BIAS OF ONLY ONE SIGN
C
  501 T = FLOAT(MPOINT)/POINTS
      X = (AMAX**2)/(3.*GAMMA*T*(1.-T))
      DFN = 3.
      DFD = POINTS/2
      CALL MDFDRE(X,DFN,DFD,P,IER)
      IF(IER .EQ. 129)THEN
          WRITE(6,*)'IER ERROR'
          STOP
      END IF
      PROBAB = 1.0 - P
      IF(PROBAB .LT. ALPHA)THEN
C
C  *****BLOCK TO OVERRIDE,IF NECESSARY,
C  *****TRUCATION POINT TO ALLOW AT LEAST TWO
C  *****PASSES TO ELIMINATE INITIAL BIAS
C  *****POINT = .25*DATA
C
C
  913 CONTINUE
          MAXPNT = IFIX(.25*NUMBAT)
          IF(MPOINT .GT. MAXPNT)THEN
              MPOINT = MAXPNT
          END IF
C
C
C  *****ITRUNC EQUALS THE TRUNCATION POINT
C
      ITRUNC = ITRUNC + MPOINT
C
C  *****BLOCK TO SEE IF THE TEST PROCEDURE HAS
C  *****TRUNCATED AN EXCESSIVE AMOUNT OF DATA
C  *****(50%) AND STILL NOT ELIMINATED INITIAL BIAS
C
      JUNK = ITRUNC*NSIZE
C
      IF(JUNK .GT. (0.5*LENTH))THEN
      IFAULT = 50
      RETURN
      END IF
C
C  *****CALCULATE NUMBER OF DATA POINTS
C  *****LEFT AND RETEST USING ONLY THESE POINTS
C
```

```
            OBS = OBS - (MPOINT*NSIZE)
            IBEGIN = IBEGIN + (MPOINT*NSIZE)
            IEND = IBEGIN + (NSIZE - 1)
            GO TO 75
        ELSE
C
C
C
C  *****SHOW FINAL TRUNCATION POINT
C  *****AND COMPUTED MEAN
C
  974 CONTINUE
      END IF
      IFAULT = 0
      RETURN
      END
C
**********************  END OF IB DETECTION SUBROUTINE  ***********************



C
C


C **   SUBROUTINE TO CALCULATE REQUIRED RUN LENGTH OF SIMULATION
C **   TO COMPARE ALTERNATIVES--A MULTIPLE RANKING PROCEDURE---
C **   CALCULATIONS BASED ON EXTENTION OF DUDEWICZ-DALAL MRP
C **   TO ADDRESS ANY STATIONARY ARMA(P,Q) PROCESS.  EXTENTION
C **   OF EXISTING PROCEDURE DONE BY J. R. WILSON AND R. T. DICKINSON
C **   UNIVERSITY OF TEXAS 1983


C
C
      SUBROUTINE MRPSUB(NUMBAT,BATCH,NEXTRA,IFAULT)
      DIMENSION BATCH(10000)


C **   THIS IS WHERE THE VALUES OF THE PARAMETERS USED BY THE
C **   MULTIPLE RANKING PROCEDURE ARE SET
C **   KPOP--EQUALS THE NUMBER OF ALTERNATIVES UNDER CONSIDERATION
C **   DSTAR--IS THE WIDTH OF THE INDIFFERENCE ZONE
C **   PCS--IS THE DESIRED PROBABILITY OF CORRECT SELECTION


C **   NMAX IS USED TO RECORD THE LARGEST SAMPLE SIZE REQUIRED
C **   DURING THIS EXPERIMENT
C###############  INITIALIZE VARIABLES  ###########################
      NMAX = 0
      DSTAR = 0.1
      PCS = .900
      KPOP = 3
```

```
C##################################################
C#####  CALL SUBROUTINE TO CALCULATE  ##############
C#####  DUDEWICZ AND DALAL H VALUE (DNDH)  #########
C
C
      CALL RNKSEL(NUMBAT,KPOP,PCS,DNDH)
C
C##################################################
C
C
C##################################################
      MID = 1
      CALL WELCH(BATCH,MID,NUMBAT,ZERO)
      VAR = ZERO
C
C##################################################
C#####   DETERMINE IF LARGER SAMPLE SIZE (NEXTRA)  ###
C#####   IS NEEDED BASED ON DUDEWICZ, RAMBERG, AND  ###
C#####   CHEN PROCEDURE.  NEXTRA .GT. NUMBAT       ###
C
C
        IDDOBS=((VAR*(DNDH**2))/(DSTAR**2))+.999999
          NEXTRA = MAX0(NUMBAT,IDDOBS)
               IF(NEXTRA .GT. NUMBAT)THEN
                  IFAULT = 100
                  RETURN
               END IF
C
C
C
      NMAX = NEXTRA
      WRITE(6,110)PCS,DSTAR,NUMBAT,DNDH,NMAX
  110 FORMAT(///,6X,'FOR THIS TEST: PCS =',F4.3,
     $'   DSTAR =',F5.2,/,6X,'   NUMBAT =',I4,'   DNDH =',F5.3,
     $'   NMAX =',I5)

C
      RETURN
      END
C


C **  SUBROUTINE TO CALCULATE THE CRITICAL DUDEWICZ-
C **  DALAL H VALUE NECESSARY TO DETERMINE REQUIRED
C **  SIMULATION RUN LENGTH

C
      SUBROUTINE RNKSEL(NO,K,PSTAR,H)
      COMMON/RSTOL/TOLF,HTOLF,NSIGD,AERR,RERR,ITMAX
      COMMON/RSCON/PI,TOLZ,BIGM
      TOLF = 1.E-6
      AERR = 1.E-8
      NSIGD = 5
      RERR = 1.E-8
      ITMAX = 500
```

```
      PI = 3.1415926535
      TOLZ = 1.E-20
      BIGM= 1.E20
      H = HO(NO,K,PSTAR,IFAULT)
      RETURN
      END
      FUNCTION HO(NNO,KK,PPSTAR,IFAULT)
      EXTERNAL GFUNC
      COMMON/RSTOL/TOLF,HTOLF,NSIGD,AERR,RERR,ITMAX
      COMMON/RSCON/PI,TOLZ,BIGM
      COMMON/RSTDST/NO,DF,K,KM1,CNORM,XPNT,A,B,PSTAR,HTEMP
      DIMENSION WK(30),PAR(1),H(1)
      DATA NDIM/1/
      NO = NNO
      K = KK
      PSTAR = PPSTAR
      HO = -BIGM
      IFAULT = 0
      DF = NO - 1
      KM1 = K - 1
      XPNT = (DF + 1.0)/2.0
      CNORM = GAMMA(XPNT)/( GAMMA(0.5*DF)*SQRT(PI*DF) )
      HTOLF = TOLF/2.0
      CALL MDSTI(HTOLF,DF,B,IER)
      IF (IER .NE. 0) THEN
         IFAULT = IER + 1000
         RETURN
      END IF
      A = -B
      H(1) = 0.0
      CALL ZSCNT(GFUNC,NSIGD,NDIM,ITMAX,PAR,H,FNORM,WK,IER)
      HO = H(1)
      IF (IER .NE. 0) THEN
      IFAULT = 2000 + IER
      END IF
      RETURN
      END
      SUBROUTINE GFUNC(H,G,NDIM,PAR)
      DIMENSION H(NDIM),G(NDIM),PAR(1)
      COMMON/RSTOL/TOLF,HTOLF,NSIGD,AERR,RERR,ITMAX
      COMMON/RSCON/PI,TOLZ,BIGM
      COMMON/RSTDST/NO,DF,K,KM1,CNORM,XPNT,A,B,PSTAR,HTEMP
      EXTERNAL SUMMND
      HTEMP = H(1)
      GTEMP = DCADRE(SUMMND,A,B,AERR,RERR,ERROR,IER)
      IF (IER.GE.100) THEN
      IFAULT = 3000 + IER
      WRITE (6,100) IFAULT
  100 FORMAT (' ***ERROR IN DCADRE = ',I4,' ***')
      STOP
      END IF
      G(1) = GTEMP - PSTAR
      RETURN
      END
      FUNCTION SUMMND(T)
```

```
      COMMON/RSTOL/TOLF,HTOLF,NSIGD,AERR,RERR,ITMAX
      COMMON/RSCON/PI,TOLZ,BIGM
      COMMON/RSTDST/NO,DF,K,KM1,CNORM,XPNT,A,B,PSTAR,HTEMP
      F1 = CNORM*( (1.0 + T*T/DF)**(-XPNT) )
      X = T + HTEMP
      CALL MDTD( ABS(X), DF, TAILS, IER)
      IF (IER .GT. 0) THEN
         IFAULT = 4000 + IER
         WRITE (6,100) IFAULT
100   FORMAT (' ***ERROR IN MDTD = ',I4,' ***')
      END IF
      F2 = 0.5 + SIGN(0.5,X)*(1.0 - TAILS)
      SUMMND = F1*( F2**KM1 )
      RETURN
      END


C ****************************  END MRPSUB  ********************************


C
C
C  ****SUBROUTINE WELCH USED TO DETERMINE VARIANCE BY USE OF SPECTRAL
C  ****DENSITY AT ZERO FREQUENCY PER ARTICLE BY HEIDELBURGER
C **  AND WELCH.  THIS SUBROUTINE CALLED BY BOTH IBSUB
C **  AND MRPSUB.
C
C
C
      SUBROUTINE WELCH (BATCH,MID,NUMBAT,ZERO)
      DIMENSION PERIOD(600),XM(6),TEMP(6),B(6,7)
     $,ANOVA(16),VARB(21),FJ(300),IWK(3200),WK(3200),V(300,6),VCV(21)
     $,NBR(6),ALFA(2),IJOB(2),IND(11),XYB(6,5),A(300,6),CHECK(2000)
      DIMENSION BATCH(10000)
      COMPLEX TRANS(600)
C
C
      NUMDAT = NUMBAT - MID + 1
      INUM = 0
      DO 10 I = MID,NUMBAT
        INUM = INUM + 1
        CHECK(INUM) = BATCH(I)
   10 CONTINUE
      NHDATA = NUMDAT/2
      NQDATA = NHDATA/2
      CALL FFTRC(CHECK,NUMDAT,TRANS,IWK,WK)
      MMM = NHDATA + 1
      DO 15 L=2,MMM
      K = L - 1
        PERIOD(K)=(CABS(TRANS(L))**2)/NUMDAT
   15 CONTINUE
      DO 30 KL=1,NQDATA
        KK =2*KL
```

```
      JJ = KK-1
      SMOTH=((PERIOD(JJ)+PERIOD(KK))/2.0)
      FJ(KL)=ALOG(SMOTH) + .270
   30 CONTINUE
      DO 50 I=1,NQDATA
          A(I,1) =   I
          A(I,2) =   I*I
          A(I,3) = I**3
      A(I,4) = I**4
      A(I,5) = I**5
          A(I,6) = FJ(I)
   50 CONTINUE
      M = 5
      IB = 6
      IND(1) = 0
      IND(2) = 0
      IND(3) = 0
      IND(4) = 0
      IND(5) = 0
      IJOB(1) = 0
      IJOB(2) = 1
      ALFA(1) = .05
      ALFA(2) = .05
      NRDIM = 300
      CALL RLSEP(A,NQDATA,M,NRDIM,ALFA,IJOB,IND,ANOVA,XYB,IB,VARB,IER)
      J = 0
      DO 2000 I=1,5
        IF (XYB(I,2) .NE. 0)THEN
           J = J + 1
           DO 2100 LL = 1,NQDATA
             V(LL,J) = LL**I
 2100      CONTINUE
        END IF
 2000 CONTINUE
      IF(J .EQ. 0)THEN
          ZERO = EXP(XYB(6,2))
          RETURN
          ELSE
      J = J + 1
      DO 2200 I = 1,NQDATA
        V(I,J) = FJ(I)
 2200 CONTINUE
      NVAR = J
      NBR(1) = NVAR
      NBR(2) = NQDATA
      NBR(3) = NQDATA
      NBR(4) = 1
      NBR(5) = 1
      NBR(6) = 1
      CALL BECOVM(V,NRDIM,NBR,TEMP,XM,VCV,IER)
      IVAR = NVAR - 1
      ALPHA = 0.05
      CALL RLMUL(VCV,XM,NQDATA,IVAR,ALPHA,ANOVA,B,IB,VARB,IER)
      UPLEFT = (B(NVAR,4)**2)/ANOVA(8)
      ADJUST = (.645*UPLEFT)/2.0
```

```
      CONE = EXP(-ADJUST)
      ZERO = CONE*(EXP(B(NVAR,1)))
      END IF
      RETURN
      END
C
```

APPENDIX F

```
C **  THIS PROGRAM IS SLAM SIMULATION USED TO MODEL AN (S,S)
C **  INVENTORY SYSTEM FOR 100 YEARS.  THE WEEKLY INVENTORY
C **  HOLDING COSTS, SHORTAGE COSTS, AND ORDERING COSTS ARE
C **  SET BY THE USER (DEFAULT VALUES ARE INCLUDED).
C **  THE MODELWILL INITIALIZE THE SYSTEM IN YEAR 1 AT AN
C **  INVENTORY LEVEL OF BIG S

C **  THE STEADY STAE PERFOMANCE MEASURE FOR COMPARING THE
C **  ALTERNATIVE REORDER POINT (LITTLES) AND THE ORDER UP
C **  TO LEVEL (IBIGS) AGAINST OTHERS IS CALCULATED BY THE
C **  INTEGRATED MULTIPLE RANKING PROCEDURE DEVELOPED BY
C **  J WILSON AND T DICKINSON AT THE UNIVERSITY OF TEXAS


C **  THE CODE WRITTEN STARTING AT SUBROUTINE NQUE MAY BE LIFTED
C **  AND USED AS A MULTIPLE RANKING PROCEDURE FOR ANY DISCRETE
C **  EVENT SIMULATION WRITEEN USING SLAM

C **  IT IS POSSIBLE WITH MINOR MODIFCATION TO USE THIS PROCEDURE
C **  ON A SIMULATION UTILIZING ANOTHER LANGUAGE


      PROGRAM MAIN(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,TAPE7,TAPE8)
      COMMON QSET(5000)
      COMMON/SCOM1/ ATRIB(100),DD(100),DDL(100),DTNOW,II,MFA,MSTOP,NCLNR
     1,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,SS(100),SSL(100),TNEXT,TNOW,XX(100)
     2,NSTRM,KOUNT,NSIZE,ALPHA,LITTLES,IBIGS,KKK,LENROW,NREP,IEXP
C
      NNSET = 5000
      NCRDR = 5
      NPRNT = 6
      NTAPE = 7
      NSTRM = 7
      KOUNT = 0
      LITTLES = 2
      IBIGS = 6
      LENROW = 100
      NREP = 16
      NSIZE = 1
      ALPHA = .1
      CALL SLAM
      STOP
      END


C **  SLAM SUBROUTINE TO SET INITAL CONDITIONS FOR THE SIMULATION


      SUBROUTINE INTLC
      COMMON/SCOM1/ ATRIB(100),DD(100),DDL(100),DTNOW,II,MFA,MSTOP,NCLNR
     1,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,SS(100),SSL(100),TNEXT,TNOW,XX(100)
     2,NSTRM,KOUNT,NSIZE,ALPHA,LITTLES,IBIGS,KKK,LENROW,NREP,IEXP
```

```
      COMMON/STOCK/IDMD(52),BUFFR(7),INVHAND,TOTALC,HC,OC,SC
C
C**** SET UP INITIAL CONDITIONS FOR MODEL
C



C
      DO 5 I=1,52
        IDMD(I) = 0
    5 CONTINUE

      TOTALC = 0

C **  SET VALUES FOR ORDERING,HOLDING AND SHORTAGE
      HC = 0.1
      OC = 0.5
      SC = 1.0


C **  INITIIALIZE INVENTORY ON HAND TO IBIGS

      INVHAND = IBIGS




C
C
C
C**** INITIALIZE KKK FOR EACH RUN
      KKK = 0



      IF(MOD(NNRUN,16) .NE. 0)THEN
           IMUL = NNRUN/16
           IEXP = IMUL + 1
           KOUNT = NNRUN - (IMUL*16)
      ELSE
           KOUNT = 16
           IEXP = NNRUN/16
      END IF
      IF (KOUNT .EQ.1)THEN
          WRITE(6,*)' EXPERIMENT NUMBER =',IEXP
      END IF
C
C
C
C
C**** POST ENTRY ON EVENT CALENDAR WITH EVENT CODE = 1, DELAY = 00.0
```

```
C
      CALL SCHDL(1,0.0,ATRIB)
      RETURN
      END

C **  SUBROUTINE TO SCHEDULE PERIODIC INVENTORY REVIEWS
C **  AND COLLECTION OF YEARLY COSTS BY THE ANALYSIS ROUTINE


      SUBROUTINE EVENT(ICODE)
C
C**** INVOKE APPROPRIATE EVENT PROCESSING ROUTINE
C
      GO TO (10,20),ICODE
C
C     ARRIVAL EVENT
   10 CALL INVENT
      RETURN
C
C       DATA EVENT
   20 CALL NQUE
      RETURN
      END
C


C **  SLAM SUBROUTINE TO CALCULATE YEAR INVENTORY COST


C
      SUBROUTINE INVENT
      COMMON/SCOM1/ ATRIB(100),DD(100),DDL(100),DTNOW,II,MFA,MSTOP,NCLNR
     1,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,SS(100),SSL(100),TNEXT,TNOW,XX(100)
     2,NSTRM,KOUNT,NSIZE,ALPHA,LITTLES,IBIGS,KKK,LENROW,NREP,IEXP
      COMMON/STOCK/IDMD(52),BUFFR(7),INVHAND,TOTALC,HC,OC,SC
      DIMENSION OCOST(52),BCOST(52),CWEEK(52),CCOST(52)

      CALL SCHDL(1,0.1,ATRIB)

      DO 10 I=1 52
      OCOST(I) = 0.0
      CCOST(I) = 0.0
      BCOST(I) = 0.0
      CWEEK(I) = 0.0
      IDMD(I) = 0
   10 CONTINUE


      XX(1) = 0.0
      TOTALC = 0.0
```

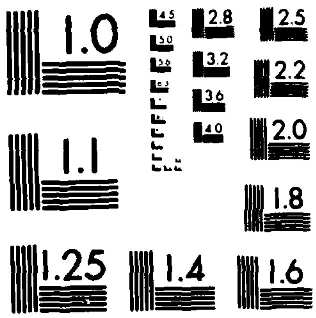
```
C **  FIGURE WEEKLY COSTS

      DO 20 II=1,52


C **  DETERMINE DEMAND

   40 IDMD(II) = UNFRM(0.,7.,NSTRM)

      IF(IDMD(II) .EQ. 7)GO TO 40
C **  UPDATA INVENTORY LEFT

      INVHAND = INVHAND - IDMD(II)


C ** IF CANNOT MEET DEMAND CHARGE SHORTAGE COST

      IF(INVHAND .LT. 0)THEN
          BCOST(II) = (-INVHAND)*SC
      ELSE
C **  IF NOT SHORTAGE COST CHARGE HOLDING COST

          CCOST(II) = INVHAND*HC

      END IF

C **  DETERMINE THE ORDER NEEDED BASED ON THIS WEEKS TRANSACTIONS

      IF(INVHAND .LT. LITTLES)THEN
           OCOST(II) = OC
            INVHAND = IBIGS
      END IF


C **   TOTAL WEEKLY COSTS



        CWEEK(II) = OCOST(II)+BCOST(II)+CCOST(II)
        TOTALC = TOTALC + CWEEK(II)
   20 CONTINUE


C **  THE SLAM VARIABLE XX(1) IS USED TO PASS THE YEARLY COST
C **  INFORMATION TO THE ANALYSIS ROUTINE
```

```
      XX(1) = TOTALC
      MMM = MMM + 1
      CALL SCHDL(2,0.0,ATRIB)
      RETURN
      END

C
C
C


C **  SUBROUTINE TO COLLECT DATA AND CALL APPROPRIATE SUBROUTINE
C **  WHEN INITIAL SIMULATION RUNS ARE COMPLETE--CALLS SUBROUTINES
C **  TO INDUCE NORMALITY(WILK), DETECT AND TRUNCATE INITIAL BIAS
C **  (IBSUB), AND CALCULATE REQUIRED SIMULATION RUN LENGTH(MRPSUB)
C **  BASED ON THIS RUN LENGTH THE ROUTINE CALCULATES
C **  STATISTIC (COMPARE) USED TO RANK ALTERNATIVES



C
C
C
      SUBROUTINE NQUE
      COMMON/SCOM1/ ATRIB(100),DD(100),DDL(100),DTNOW,II,MFA,MSTOP,NCLNR
     1,NCRDR,NPRNT,NNRUN,NNSET,NTAPE,SS(100),SSL(100),TNEXT,TNOW,XX(100)
     2,NSTRM,KOUNT,NSIZE,ALPHA,LITTLES,IBIGS,KKK,LENROW,NREP,IEXP
      DIMENSION BUFFR(7),ADD(15000),DATA(16,100),BATCH(5000)
     $,DNORMAL(5000),ARRAYC(100)

      KKK = KKK + 1
      IF ( IFAULT .EQ. 50) GO TO 1990


      IF(KKK .GT. LENROW)GO TO 7
      IBDATA = LENROW
      DATA(KOUNT,KKK) = XX(1)
      IF(KOUNT.EQ.NREP.AND.KKK.EQ.LENROW)THEN
          LLL = 0
          CALL SWSUB(DATA,NREP,NSIZE,IFAULT)


      IF (NSIZE .GT. 3 .AND. IFAULT .NE. 10)THEN
          WRITE(6,*)' PASSED SW TEST BUT NSIZE TOO BIG'
          NSIZE = 3
      END IF
      IF (IFAULT .EQ. 10)THEN
         WRITE(6,*)' PROBLEMS NSIZE TO BIG'
         WRITE(6,*)'CONTINUE WITH NSIZE = 3'
         NSIZE = 3
      END IF
```

AD-A133 649 A MULTIPLE RANKING PROCEDURE ADAPTED TO DISCRETE-EVENT SIMULATION(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH R T DICKINSON DEC 83 AFIT/CI/NR-83-47D 3/3

UNCLASSIFIED F/G 12/1 NL

END
DATE
FILMED
11 83
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963 A

REFERENCES

Andrews, R. W. and Schriber, T. J. "Two ARMA-Based Confidence-Interval Procedures for the Analysis of Simulation Output," School of Business University of Michigan, 1982.

Bahadur, R. R. "On the Problem in the Theory of K Populations," Annuals of Mathematical Statistics, 21 (1950), 362-375.

Barr, D. R. and Rizvi, M. H. "An Introduction to Ranking and Selection Procedures," Journal American Statistical Association, 61 (1966), 640-646.

Bechhofer, R. E. "A Single Sample Multiple Decision Procedure for Ranking Means of Normal Populations with Known Variances," Annals of Mathematical Statistics, 25 (1954), 16-39.

Bechhofer, R. E. "A Sequential Multiple Decision Procedure for Selecting the Best One of Several Normal Populations with a Common Variance, and its Use with Various Experimental Designs," Biometrics, 18 (1958), 408-429.

Bechhofer, R. E. and Blumental, S. "A Sequential Multiple-Decision Procedure for Selecting the Best One of Several Normal Populations with a Common Unknown Variance, II: Monte Carlo Sampling Results and New Computing Formulas," Biometrics, 18 (1962), 52-67.

Bechhofer, R. E., Dunnett, C. W., and Sobel M. "A Two-Sample Multiple Decision Procedure for Ranking Means of Normal Populations with a Common Unknown Variance," Biometrika, 41 (1954), 170-176.

Bloomfield, P. Fourier Analysis of Time Series: An Introduction John Wiley and Sons, Inc. New York, 1976.

Bowker, A. H. and Lieberman, G. J. Engineering Statistics Prentice-Hall, Inc. Englewood Cliffs, 1972.

Box, G. E. P. and Jenkins, G. M. Time Series Analysis: Forecasting and Control. Holden-Day, San Francisco, 1970.

Brillinger, D. R. Time Series: Data Analysis and Theory Holt, Rinehart and Winston, Inc. New York, 1975.

Chambers, M. L. and Jarrett, P. "Use of Double Sampling for Selecting Best Population," Biometrika, 41 (1964), 49-64.

Conway, R. W. "Some Tactical Problems in Digital Simulation," Management Science, 10 (1963), 47-61.

Cooley, J. W., Lewis, P. A. W., and Welch, P. D. "The Fast Fourier Transform Algorithm: Programming Considerations in the Calculations of Sine, Cosine, and La Place Transforms." Journal of Sound Vibration, 12 (1970a), 315-337.

Cooley, J. W., Lewis, P. A. W., and Welch, P. D. "The Application of the Fast Fourier Transform Algorithm to the Estimation of Spectra and Cross-Spectra," Journal of Sound Vibration, 12 (1970b), 339-352.

De Boor, C. "CADRE: An Algorithm for Numerical Quadrature," Mathematical Software, ed. J. R. Rice, Academic Press, New York, 1971.

De Branges, C. The Shapiro-Wilk Test for Normality. Department of Statistics, Purdue University, West Lafayette, IN., 1974.

Draper, N. R. and Smith H. Applied Regression Analysis John Wiley and Sons, Inc. New York, 1967.

Dudewicz, E. J. "An Approximation to the Sample Size in Selection Problems," Annuals of Mathematical Statistics, 40 (1969), 492-497.

Dudewicz, E. J. "Nonexistance of a Single-Sample Selection Procedure Whose P(CS) is Independent of the Variances," South African Statistical Journal, 5 (1971), 37-39.

Dudewicz, E. J. "Statistics in Simulation: How to Design for Selecting the Best Alternative," Proceedings of the 1976 Winter Simulation Conference, (1976), 67-81.

Dudewicz, E. J. and Dalal, S. B. "Allocation of Observations in Ranking and Selection with Unequal Variances," Sankhya, Series B, 37 (1975), 28-88.

Dudewicz, E. J., Ramberg, J. S., and Chen, H. J. "New Tables of Multiple Comparison With a Control (Unknown Variances)," Biometrische Zeitchrift, 17 (1975), 13-26.

Dudewicz, E. J. and Zaino, N. A. "Allowance for Correlation in Setting Simulation Run-Length Via Ranking-And-Selection Procedures," TIMS Studies in the Management Sciences, 7 (1977), 51-61.

Duket, S. D., and Pritsker, A. A. B. "Examination of Simulation Output Using Spectral Methods," Mathematics and Computers in Simulation, 20 (1978), 53-60.

Dunnett, C. W. "On Selecting the Largest of K Normal Population Means," Journal of the Royal Statistical Society Series, B, (1960), 1-40.

Dunnett, C. W. and Sobel, M. "A Bivarate Generalization of Student's t - Distribution with Tables for Certain Special Cases," Biometrika, 41 (1954), 153-169.

Fishman, G. S. "Estimating Sample Size in Computing Simulation Experiments," Management Science, 18 (1971), 21-38.

Fishman, G. S. Principles of Discrete Event Simulation Wiley Interscience, New York, 1978.

Fishman, G. S. Concepts and Methods in Discrete Event Digital Simulation Wiley Interservice, New York, 1973.

Fishman, G. S., and Kiviat, P. J. "Spectral Analysis of Time Series Generated by Simulation," Management Science, 13 (1967), 525-557.

Gross, D. and Harris, C. M. Fundamentals of Queueing Theory John Wiley and Sons, Inc. New York, 1974.

Gupta, S. S. "On a Decision Rule for a Problem in Ranking Means," Series No. 150, Institute of Statistics, University of North Carolina, Chapel Hill, N. C., (1956).

Gupta, S. S. "On Some Multiple Decisions (Selection and Ranking) Rules," Technometrics, 7 (1965), 225-245.

Gupta, S. S. and Hsu, J. C. "Subset Selection Procedures with Special Reference to the Analysis of Two-Way Layout: Application to Motor-Vehicle Fatality Data," Proceedings of the 1977 Winter Simulation Conference, (1977), 81-85.

Gupta, S. S. and Panchapakesan, S. "Selection and Ranking Procedures," The Design of Computer Simulation Experiments edited by T. H. Naylor, Duke University Press, (1969), 132-160.

Hadley, G. and Whitin, T. M. Analysis of Inventory Systems Prentice-Hall, Inc., Englewood Cliffs, 1963.

Heidelberger, P., and Welch, P. D. "A Spectral Method for Confidence Interval Generation and Run Length Control in Simulation" Communications of the ACM, 24 (1981a), 233-245.

Heidelberger, P., and Welch, P. D. "Adaptive Spectral Methods for Simulation Output Analysis," IBM Journal of Research and Development, 25 (1981b), 860-876.

Heidelberger, P. and Welch, P. D. "Simulation Run Length Control in the Presence of an Initial Transient," IBM Technical Report 41425, 1982.

Hillier, F. S. and Lieberman, G. J. Introduction to Operations Research Holden-Day, Inc. San Francisco, 1980.

Hoffman, S. "An Integrated Model of Drilling Vessel Operations," Unpublished Masters Thesis, The University of Texas at Austin, 1982.

Iglehart, D. L. "Simulating Stable Stochastic Systems, VII: Selecting the Best System," TIMS Studies in the Management Sciences, 7 (1977), 37-49.

James, R. L. "Routing Control in a Packet-Switched Network," Unpublished Dissertation, University of Texas at Austin, 1981.

Jenkins, G. M. and Watts, D. G. Spectral Analysis and Its Applications Holden-Day, Inc. San Francisco, 1968.

Jucker, J. V. and Gomez, J. G. "Policy-Comparing Simulation Experiments: Design and Analysis," Decision Science, 6 (1975), 631-645.

Kleijnen, J. P. C. Statistical Techniques in Simulation, Part II Marcel Dekker, Inc., New York, 1975.

Kleijnen, J. P. C. and Naylor, T. H. "The Use of Multiple Ranking Procedures to Analyze Simulations of Business and Economic Systems," Proceedings of the Business and Economic Statistics Section, Annual Meeting of the American Statistical Association New York, 1969, 605-615.

Kleijnen, J. P. C., Naylor, T. H., and Seeks, T. G. "The Use of Multiple Ranking Procedures to Analyze Simulations of Management Systems: A Tutorial" Management Science 18 (1972), B245-B257.

Law, A. M. and Kelton, W. D. Simulation Modeling and Analysis McGraw-Hill Book Company, New York, 1982.

Mosteller, Frederick. "A K-sample Slippage Test for an Extreme Population" Annuals of Mathematical Statistics, 21 (1948), 58-65.

Naylor, T. H., Burdick, D. S., and Sasser, W. E. "Computer Simulation Experiments with Economic Systems: The Problem of Experimental Design" Journal of the American Statistical Association, 62 (1967), 1315-1337.

Naylor, T. H. and Chu, K. "A Dynamic Model of the Firm," Management Science, 11 (1965), 736-750.

Naylor, T. H. et al. Computer Simulation Techniques John Wiley and Sons, Inc. New York, 1966.

Naylor, T. H., Wertz, K., and Wonnacott, T. "Methods for Analyzing Data From Computer Simulation Experiments," Communications of the ACM, 10 (1967), 703-810.

Naylor, T. H., Werta K., and Wonnacott, T. H. "Spectral Analysis of Data Generated by Simulation Experiments with Economic Models," Ecometrics, 37 (1969), 333-352.

Nelson, C. R. Applied Time Series Analysis For Managerial Forecasting Holden-Day, Inc. San Francisco, 1973.

Paulson, E. "A Sequential Procedure for Selecting the Population with the Largest Mean from K Normal Populations," Annuals of Mathematical Statistics, 35 (1964), 174-180.

Pritsker, A. A. B. and Pegden, C. D. Introduction to Simulation and SLAM Halsted Press, West Lafayette, 1979.

Sasser, W. E., et al. "The Application of Sequential Sampling to Simulation: An Example Inventory Model," Communications of the ACM, 13 (1970), 287-296.

Schruben, L. W. "Controlling Initialization Bias in Simulation Experiments," Cornell University School of Operations Research Technical Report 391, 1979.

Schruben, L. W. "Coverage Function for Interval Estimators of Simulation Responses," Management Science, 26 (1980), 18-27.

Schruben, L. W. "Detecting Initialization Bias in Simulation Output," Operations Research, 30 (1982), 569-590.

Shapiro, S. S. and Francia, R. S. "An Approximate Analysis of Variance Test for Normality," Journal of the American Statistical Association, 67 (1972), 215-216.

Shapiro, S. S. and Wilk, M. B. "An Analysis of Variance Test for Normality (Complete Samples)," Biometrika, 52 (1965), 591-611.

Shapiro, S. S., Wilk, M. B., and Chen, H. J. "A Comparative Study of Various Tests for Normality," Journal of the American Statistical Association, 63 (1968), 1343-1372.

Somerville, P. E. "Optimum Sample Size for a Problem in Choosing the Population with the Largest Mean," Journal of the American Statistical Association, 65 (1970), 763-875.

Starr, N. "The Performance of a Sequential Procedure for the Fixed-width Interval Estimation of the Mean," Annuals of Mathematical Statistics, 37 (1966), 36-50.

Steudel, H. J., Pandit, S. M., Wu, S. M. "Interpretation of Dispatching Policies on Queue Behavior via Simulation and Time Series Analysis," AIIE Transactions, 10 (1978), 292-298.

Steudel, H. J., and Wu, S. M. "A Time Series Approach to Queuing Systems with Applications for Modeling Job-shop In-Process Inventories," Management Science, 23 (1977), 745-855.

Watts, D. "Time Series Analysis," The Design of Computer Simulation Experiments edited by T. H. Naylor, Duke University Press, 1969, 165-179.

Weisburg, S. and Bingham, C. "An Approximate Analysis of Variance Test for Non-Normality Suitable for Machine Calculation," Technometrics, 17 (1975), 133-134.

Wilson, J. R. and Pritsker, A. A. B. "A Survey of Research on the Simulation Startup Problem," Simulation, 30 (1978a), 55-58.

Wilson, J. R. and Pritsker, A. A. B. "Evaluation of Startup Policies in Simulation Experiments," Simulation, 31 (1978b), 79-89.

Wolfe, P. "The Secant Method for Solving Simultaneous Nonlinear Equations," Communications of the ACM, 2 (1959), 12-13.

VITA

Robert Timothy Dickinson was born in Columbus, Ohio, on May 5, 1945, the son of Alice and Hal Dickinson. He received a Bachelor of Science degree in Electrical Engineering from Ohio University in June 1968. Upon graduation, he was commissioned in the U. S. Air Force and assigned as a test engineer to Hill AFB, Utah. While assigned there he attended the University of Utah and was awarded a Masters of Engineering Administration in June 1972. Subsequent Air Force assignments included: Development Engineer at the Air Force Plant Representative's Office - Hughes Aircraft Co., Fullerton, California; Chief of the Operations Branch, 3246th Test Wing, Eglin AFB, Florida; and Program Control Officer, Operational Range Equipment Program Office, Eglin AFB, Florida. In 1980, he was selected by the Air Force to attend graduate school in the field of Operations Research. He entered the Graduate School of The University of Texas at Austin in September of that year. He is married to Joyce Elaine Jones of Pittsburgh, Pennsylvania. They have two daughters, Brooke Ellen and Adrienne Alice, born in 1969 and 1972, respectively.

Permanent Address: 3248 Southfield Drive
Xenia, Ohio 45385

This dissertation was typed by Mrs. Margaret Jensen.

LMED

8